**A project report submitted in partial fulfilment for the degree of**

**BSc (Hons) Computer Games Development**

**School of Psychology and Computer Science**

**University of Central Lancashire**

Cross stitch pattern generator

Lili Veszeli

**12 April 2021**

# Abstract

Remove any text highlighted in yellow before submission.

You are aiming for no more than 50 pages of report content, this count starts at Chapter 1 and does not includes your references or appendices.

The abstract is the summary of the project report within one page (aim for about 500 words). Unnumbered chapter headings, as above, are entered using the ‘Heading (Unnumbered)’ style, which automatically starts a new page.

This template starts the page numbering at the foot of this page. That is, the first page does not have a number.

It is suggested that the abstract be structured as follows:

* Problem: What you tackled, and why this needed a solution
* Objectives: What you set out to achieve, and how this addressed the problem
* Methodology: How you went about solving the problem
* Achievements: What you managed to achieve, and how far it meets your objectives.

IMPORTANT – your abstract is a SUMMARY of the content of this report. It’s like a synopsis of a movie, with spoilers. Write it in the present tense. For example:

This project report discusses the process of implementing xxxxx. This report reviews, amongst other topics the current literature available on the use of xxxxx in xxxxx along with how the project has been planned, designed and developed.

Very formal

Describe the creation of the rceation of across stitch pattern. Quantization was used, wpf was used to show this pattern. Was successful, managed to achieve the goal.

# Attestation

I understand the nature of plagiarism, and I am aware of the University’s policy on this.

I certify that this document reports original work by me during my University project.

Signature: Lili Veszeli

Date: 01/04/2021

# Acknowledgements

Acknowledge anyone who has helped you in your work such as your supervisor, technical support staff, fellow students, or external organisations. Acknowledge the source of any work that is not your own.

Laurent Noel, Sam Connolly, John Hooson

# Table of Contents

The table of contents below is automatically generated from the paragraphs of style ‘Heading 1’ to ‘Heading 3’ and ‘Heading (Unnumbered)’. To update this after revisions, right-click in the table and choose ‘Update Field’ for the entire table. It is also possible to left-click the table and press F9 to update the entire table. Delete this paragraph and the one below before submission.

Sometimes Word does not automatically update references to other parts of the document, such as captions, Table of Contents and cross-references. To update everything, select the whole document by pressing Ctrl+A and then either press F9 or right-click anywhere and click on ‘Update Field’. Confirm all prompts by selecting ‘Update entire table’ and clicking OK. **Sometimes the font changes on update so check it is the correct font face before submitting your work**.

[Abstract i](#_Toc70602134)

[Attestation ii](#_Toc70602135)

[Acknowledgements iii](#_Toc70602136)

[Table of Contents iv](#_Toc70602137)

[List of Figures ix](#_Toc70602138)

[List of Tables x](#_Toc70602139)

[List of Listings xi](#_Toc70602140)

[1 Introduction 1](#_Toc70602141)

[2 Background and Related Work 2](#_Toc70602142)

[2.1 Introduction 2](#_Toc70602143)

[2.2 Cross Stitch History and Basics 2](#_Toc70602144)

[2.3 Image Processing 3](#_Toc70602145)

[2.3.1 Detail Preserving Resampling 4](#_Toc70602146)

[2.3.2 Posterization 5](#_Toc70602147)

[2.4 Application Platforms for Consideration 5](#_Toc70602148)

[2.4.1 .NET Framework 5](#_Toc70602149)

[2.4.2 Windows Forms 6](#_Toc70602150)

[2.4.3 WPF 6](#_Toc70602151)

[2.4.4 Comparison 7](#_Toc70602152)

[2.5 User Interface 7](#_Toc70602153)

[2.5.1 Colour 8](#_Toc70602154)

[2.6 Existing Cross Stitch Software 9](#_Toc70602155)

[2.7 Summary 10](#_Toc70602156)

[3 Project Planning 11](#_Toc70602157)

[3.1 Introduction 11](#_Toc70602158)

[3.2 Methodology 11](#_Toc70602159)

[3.3 Requirements 12](#_Toc70602160)

[3.3.1 Must 12](#_Toc70602161)

[3.3.2 Should 13](#_Toc70602162)

[3.3.3 Could 13](#_Toc70602163)

[3.3.4 Won’t 13](#_Toc70602164)

[3.4 Potential Solutions and Tools and Techniques 13](#_Toc70602165)

[3.4.1 Application platforms 13](#_Toc70602166)

[3.4.2 Programming languages 14](#_Toc70602167)

[3.4.3 Other languages/Data languages 15](#_Toc70602168)

[3.4.4 Other tools 15](#_Toc70602169)

[3.5 Legal, Social, and Ethical Issues 16](#_Toc70602170)

[3.5.1 Ethical Issues 16](#_Toc70602171)

[3.5.2 Accessibility 17](#_Toc70602172)

[3.5.3 Legal Issues 17](#_Toc70602173)

[3.6 Summary 18](#_Toc70602174)

[4 Design 19](#_Toc70602175)

[4.1 Introduction 19](#_Toc70602176)

[4.2 Software design 19](#_Toc70602177)

[4.2.1 MVVM Pattern 19](#_Toc70602178)

[4.3 System Design 20](#_Toc70602179)

[4.4 User Interface Design 20](#_Toc70602180)

[4.5 Summary 23](#_Toc70602181)

[5 Implementation 24](#_Toc70602182)

[5.1 Introduction 24](#_Toc70602183)

[5.2 Pattern creation 24](#_Toc70602184)

[5.2.1 Downsampling 24](#_Toc70602185)

[5.2.2 Colour quantization 24](#_Toc70602186)

[5.2.3 DMC Thread colours 24](#_Toc70602187)

[5.3 Pattern editing 25](#_Toc70602188)

[5.4 Pattern symbols 25](#_Toc70602189)

[5.5 Preview 25](#_Toc70602190)

[5.5.1 Colour blending 25](#_Toc70602191)

[5.5.2 Shaders in WPF 25](#_Toc70602192)

[5.6 User Interface 25](#_Toc70602193)

[5.6.1 Undo/Redo functionality 25](#_Toc70602194)

[5.7 Summary 26](#_Toc70602195)

[6 Testing 27](#_Toc70602196)

[6.1 Introduction 27](#_Toc70602197)

[6.2 The Importance of Testing / General? 27](#_Toc70602198)

[6.3 Unit Testing 27](#_Toc70602199)

[6.4 Non-Functional Testing 28](#_Toc70602200)

[6.5 User Testing 28](#_Toc70602201)

[6.6 Automation Testing 28](#_Toc70602202)

[6.7 Evaluation 29](#_Toc70602203)

[6.8 Summary 29](#_Toc70602204)

[7 Evaluation, Conclusions and Future Work 30](#_Toc70602205)

[7.1 Project Objectives 30](#_Toc70602206)

[7.2 Self-Evaluation 30](#_Toc70602207)

[7.3 Project Evaluation 30](#_Toc70602208)

[7.4 Applicability of Findings to the Commercial World 31](#_Toc70602209)

[7.5 Conclusions 32](#_Toc70602210)

[7.6 Future Work 32](#_Toc70602211)

[7.6.1 Improvement of existing features 32](#_Toc70602212)

[7.6.2 Additional Features 32](#_Toc70602213)

[References 34](#_Toc70602214)

[Appendix 1 – Project Proposal 40](#_Toc70602215)

[Computing Degree Project Proposal 40](#_Toc70602216)

[Project Context 40](#_Toc70602217)

[Specific Objectives 40](#_Toc70602218)

[Potential Ethical or Legal Issues 40](#_Toc70602219)

[Resources 40](#_Toc70602220)

[Potential Commercial Considerations - Estimated costs and benefits 41](#_Toc70602221)

[Proposed Approach 41](#_Toc70602222)

[References 42](#_Toc70602223)

[Appendix 2 – Technical Plan 43](#_Toc70602224)

[Computing Project Technical Plan 43](#_Toc70602225)

[Title 43](#_Toc70602226)

[Summary 43](#_Toc70602227)

[Deliverables 43](#_Toc70602228)

[Constraints 43](#_Toc70602229)

[Key Problems 44](#_Toc70602230)

[System and Work Outline 44](#_Toc70602231)

[Risk Analysis 46](#_Toc70602250)

[Employability Contribution 48](#_Toc70602263)

[References 48](#_Toc70602264)

[Appendix 3 – Software Licenses 49](#_Toc70602265)

[MIT License 49](#_Toc70602266)

[Apache License 49](#_Toc70602267)

[GNU LGPL v3 License 55](#_Toc70602268)

# List of Figures

Similarly, you can automatically generate a list of ‘Figures’. Right-click a figure (e.g. image or diagram) and add a caption labelled ‘Figure’ and ‘below selected item’. To update this after revisions, right-click in this table and choose Update Field (or use F9) and then choose to update the entire table. Delete this paragraph before submission.

[*Figure 1 - Example of a cross stitch on aida (RTO, n.d.).* 2](#_Toc70602269)

[*Figure 2 - Simple cross stitch pattern (Fitzgerald, 2017)* 3](#_Toc70602270)

[*Figure 3 – The way a pattern is converted to stitches (Biedl et al., 2005)* 3](#_Toc70602271)

[*Figure 4- Box filter (Parthipan, 2017).* 4](#_Toc70602272)

[*Figure 5 - Superpixels and final result (Gertsner et al., 2012).* 5](#_Toc70602273)

[*Figure 6 - Posterized image (Kwon & Chien, 2011).* 5](#_Toc70602274)

[Figure 7 - *.NET Framework (Soumyasch, 2007).* 6](#_Toc70602275)

[Figure 8 - Photoshop User Interface (GCFGlobal, n.d.) 8](#_Toc70602276)

[Figure 9 - Colour combinations (Barševska & Rakele, 2019). 9](https://d.docs.live.net/55590c8f49fa52ac/Documents/GitHub/project/CO3808%20ProjectReportDraftLiliVeszeli.docx#_Toc70602277)

[Figure 10 *- Painting and cross stitch. Pattern created in WinStitch (Batho, 2014).* 10](#_Toc70602278)

[Figure 11 -Waterfall method (Hughey, 2009). 11](#_Toc70602279)

[Figure 12 - Iterative Agile methodology (Hazevytch, 2020). 12](#_Toc70602280)

[Figure 13 - GitHub Desktop. 16](#_Toc70602281)

[Figure 14 - Model-View-ViewModel components. 19](#_Toc70602282)

[Figure 15 - Storyboard. 21](#_Toc70602283)

[Figure 16 - Paint Tool Sai UI. 21](#_Toc70602284)

[Figure 17 - WinStitch UI. 22](#_Toc70602285)

[Figure 18 - Flow Stitch UI. 22](#_Toc70602286)

[Figure 19 - Ideal test automation pyramid (Palamarchuk, 2015). 28](#_Toc70602287)

# List of Tables

Similarly, you can automatically generate a list of ‘Tables’. Select a table, right-click it and add a caption labelled ‘Table’ and ‘above selected item’. To update this after revisions, right-click in this table and choose Update Field (or use F9) and then choose to update the entire table. Delete this paragraph before submission.

[Table 1 - Test Results 28](#_Toc70602292)

# List of Listings

[Listing 1 – [DMCColours.cs] DMC class. 24](#_Toc70602289)

# Introduction

For editorial consistency, it is important to use Word styles properly. Word 2003 onwards has so-called ‘quick styles’. If the styles referred to below are not visible on the Home ribbon in the Styles category, choose ‘Apply Styles’ from the down arrow at the bottom right of the Styles category. Styles can then be applied from the drop-down box. To make a style visible as a quick style, choose Apply Styles, then click Styles (the AA icon) or use ‘Alt + Ctrl + Shift + S’, then right-click on the style and then ‘Add to Style Gallery’.

Chapters are entered using the ‘Heading 1’ paragraph style. The Heading 1 style automatically moves to the start of a new page and supplies the next chapter number. Pressing enter on a ‘Heading 1’ heading automatically inserts a ‘Heading 2’ heading underneath.

There should not be any text between a parent heading and its first sub-heading. For example, when you want to write an introductory section for the following sections, give that introductory section an own ‘Introduction’ heading instead of writing it between the heading preceding these sections and the first sub-heading.

As an example: This comment text is incorrectly placed between a ‘Heading 1’ (‘Introduction’) and a ‘Heading 2’ (‘Background and Context’).

Most text uses the ‘Normal Project Body’ paragraph style with 10-point Calibri, 1.5-line spacing, single-sided pages.

In general, use the default spacing that headings and paragraphs give you. Avoid using new-lines or spaces to format text. If you need to use quotes, preferably use single curly quotes ‘…’. If you wish to emphasise something, use the ‘Emphasis’ style. In addition, also a ‘Strong’ style is preconfigured.

Remember to Save frequently while you are working! Check that AutoSaving is enabled under options -> save -> 'Save AutoRecover information every 5 minutes'.

# Background and Related Work

* 1. Introduction

The popularity of arts and crafts is increasing among all age groups. Textile based hobbies, like knitting and embroidery are the 2nd most popular craft in the UK (Mintel, 2020). This statistic suggests that there is a market for an arts and crafts program. The objective of the project is to create a software that converts a picture into a cross stitch pattern. The user will be able to edit the pattern and customize it using different features of the program. This chapter reviews relevant literature about the topics in the project. It explains the basics of cross stitch, to gain an understanding of the fundamentals, and existing software. Another theme is about different methods for image downsampling, which will help to create the pattern. Different application platforms are evaluated, and UI design practices are explored.

* 1. Cross Stitch History and Basics

For the purposes of this project a brief introduction to cross stitch is essential in order to understand how to implement the process digitally. Cross stitching is a type of embroidery stitch art (Setiabudi et al., 2017). A picture is created using colourful thread and little cross shaped stitches on fabric. Usually, two strands of cotton thread are used (Dyer, 1997). The stitching is done on fabric that has small holes: aida or evenweave (see Figure 1). The colour range of the threads can be vast, the most widely used is made by DMC. There are only a few types of stitches since the final result should look uniform. The simple cross is used most of the time, however, a half stitch, quarter stitch, three-quarter stitch and backstitching is also used occasionally (Atkinson & Roberts, 1999). Considering the different types of stitches used in this art form is important because they bear a close resemblance to the way that images are displayed on a computer screen because they both appear to be made of little squares (pixels).

A picture containing food, sushi

Description automatically generated

*Figure 1 - Example of a cross stitch on aida (RTO, n.d.).*

This art-form originated in Asia, and the oldest cross stitch dates back to 850 B.C. It became more popular in the Victorian era. Then, in 1980 cross stitch re-emerged again, and became how we know it today. It is one of the most popular type of needlework throughout the world (Leslie, 2007). It is a soothing and therapeutic hobby, and could help improve mental health (Hohmann, 2020).

A pattern is needed to make a counted cross stitch (see Figure 2). A pattern consists of a grid with colourful squares (Biedl et al., 2005). The squares signal the position and colour of the stitch, like in Figure 3 (Atkinson & Roberts, 1999). The size of a pattern is measured by the number of stitches across and down.

Diagram

Description automatically generated

*Figure 2 - Simple cross stitch pattern (Fitzgerald, 2017)*

*Diagram

Description automatically generated*

*Figure 3 – The way a pattern is converted to stitches (Biedl et al., 2005)*

* 1. Image Processing

Digital image processing means manipulation of an image with a computer. The cross stitch program will do exactly this: manipulating a photo into a pixelated version by compressing it. This section overviews some techniques and previous solutions to help with achieving the desired effect. Image compression is an application area of image processing. An image has to be resampled in order to upscale it, downscale it or rotate it. Downsampling is a widely used image operation. It is used to reduce the storage requirements (Youssef, 1999). However, it won’t be used in this project for that reason, but to get the pixelated look.

An image is stored in a bitmap. It is defined by the number of pixels and the information contained in them. Image scaling means to “re-sample a two-dimensional function on a new sampling grid” (Parthipan, 2017). Several algorithms can be used in image scaling. The simplest and fastest one is the nearest neighbour method, where the value of the new point becomes the value of the closest input point. Another method is linear interpolation, where the new point is interpolated between the two closest old points. Bilinear interpolation calculates the new point from the weighted average of the four closest input points. These might produce undesired effects like aliasing. A more sophisticated algorithm is bicubic interpolation. It samples the closest 4 by 4 pixels, so 16 pixels, and interpolates between them. This is the standard for most image editing software (Patil, 2018; Parker et al., 1983).

One of the simplest and most efficient downsampling algorithm is the box filter. It is a linear filter algorithm. This is considered to be used in the basic version of the program. The way it works is that each pixel in the target image is the average of the pixel values in a square from the source image. It is easier to understand with a diagram, see *Figure 4*.

A picture containing table

Description automatically generated

*Figure 4- Box filter (Parthipan, 2017).*

This is a simple way of converting a picture into bigger squares like on a pattern, but this algorithm can easily miss important details, and the resulting image/pattern might not be recognizable. Other research tries to solve this problem by using more complex algorithms.

* + 1. Detail Preserving Resampling

This algorithm by Gertsner et al. (2012) converts high resolution images into pixel art style ones with reduced colour palettes. It converts faces and other detailed images much more accurately than a regular downsampling algorithm. Pixel art is very close to a cross stitch pattern, it almost looks identical without the grid, so this article can provide great ideas to approach the problem.

Each pixel needs to be carefully placed, so that it accurately represents the original image. Areas such as the eyes and mouth are especially difficult to get right, however, it’s achievable by using “superpixels”, see *Figure 5*. The algorithm uses an iterative process. First, the superpixels are initialized in a grid, and then each original pixel is assigned to the closest superpixel. The palette is also set to an average colour. Then, the iteration starts, the superpixels are refined and associated with colours in the palette, then palette is also refined, and expanded (Gertsner et al., 2012). Unfortunately, it doesn’t allow users to pick the palette colours, so that has to be solved in another way in the program.

A picture containing text

Description automatically generated

*Figure 5 - Superpixels and final result (Gertsner et al., 2012).*

* + 1. Posterization

Posterization reduces the number of tones used in an image. There is no gradient between the areas and usually vivid colours are used (Afifi, 2018; Kwon & Chien, 2011), see *Figure 6*. This algorithm can be useful for the program since it also needs to reduce the number of colours from the original image.

First, the algorithm of Afifi (2018) removes small details from the image, then applies bilateral filtering. After the image is prepared for quantization (the process to reduce the number of colours in an image to a limited palette) (Orchard & Bouman, 1991), the pixels are sorted into ‘bright’, ‘grey’, and ‘dark’ categories using fuzzy logic.

A picture containing wall, person, clothing, suit

Description automatically generatedA picture containing text, suit

Description automatically generated

*Figure 6 - Posterized image (Kwon & Chien, 2011).*

* 1. Application Platforms for Consideration
     1. .NET Framework

A suitable application platform needs to be selected for the software. This desktop application will be Windows based, so the Microsoft .NET Framework will be used. Figure 7 illustrates the .NET stack. This technology supports running Windows and Web apps (Microsoft Docs., 2020). Both WPF and Windows Forms are part of the framework, however, they are best suited for different types of applications (Misra, 2016). Both of these platforms are evaluated and considered for the project below.

Graphical user interface, application

Description automatically generated

Figure 7 - *.NET Framework (Soumyasch, 2007).*

* + 1. Windows Forms

Windows Forms was released by Microsoft in 2002 as part of the .NET framework. This greatly influenced how Windows applications are written, explains Griffiths and Adams (2003). Before Windows Forms developers could only use Win32 to make Windows applications, this is lower level and does not have a GUI (Graphical User Interface). By increasing the level of abstraction, a higher-level object-oriented API was created. This makes it much simpler to develop an application, allowing to concentrate on the task rather than the low-level details. In Visual Studio developers can make use of the Windows Form Designer, where they can drag and drop controls into the UI. WinForms is event driven meaning when the user interacts with the interface, for example, clicking a button, an event occurs. The application processes these events with the help of event handlers, which are programmed in C# or other high-level languages (Microsoft Docs., 2017).

* + 1. WPF

WPF (Windows Presentation Foundation) is a Graphical User Interface framework (Misra, 2016). It was released in 2006 with the new .NET 3.0 framework (Xu, 2010). The expectations for user interfaces were increasing, so new technology was needed, according to Nathan (2010). The user interface needed to be separated from the implementation, so that programmers and designers could work on the application without relying on one another. As an answer to this Microsoft released WPF. The user interface design is done in XAML (Extensible Application Markup Language), completely independent of the code. XAML is an XML based markup language designed for WPF (Microsoft Docs., 2016). This way the development is more efficient, and the cost is reduced. It is also easy to understand for designers. It is much more powerful than Windows Forms since it supports documents, multimedia, 2D and 3D graphics and animation (Xu, 2010). Before it would have required several different technologies to make an application containing all these. WPF is built on Direct3D, so it can provide high performance graphics.

* + 1. Comparison

The decision about which platform to use depends on the requirements of the application. WPF allows sophisticated UI design, while in WinForms the UI is not as customizable. UI is an important part of the program, so extensive customizability is preferred. WPF supports multimedia and graphics, while WinForms would need a third-party control to do the same. The cross stitch program mainly deals with images, so support for that is essential. WinForms uses less memory than WPF, however, memory is not a primary concern. WPF might have a steeper learning curve, and has less documentation, but it still proves to be the right choice for developing the application (Misra, 2016; Microsoft Docs., 2019).

* 1. User Interface

A user interface allows the user to interact with the program (Galitz, 2007). It is part of the field of Human Computer Interaction. This topic needs to be researched in order to be able to design a good UI for the program. A good UI means one that has a well-functioning input and output mechanism and lets the user accomplish tasks without putting any obstacles in their way (Gunderloy, 2006).

The most widely used type of UI is a Graphical User Interface (GUI). The first successful system using it was released by Apple in 1984. It includes lots of visible clues (buttons, windows, mouse) and real-world analogies (like dragging a file into a trash can meaning deleting it) (MacDonald, 2008).

The main aim for it is to be intuitive and logical. It is also important to be consistent within the application and with other popular software (like Word). The reason for this is that users already learnt the layout and menu organization of them, and it would slow them down by having to learn a completely new UI (Gunderloy, 2006). The UI of the cross stitch program will be similar to widely used drawing programs like Photoshop (see **Error! Reference source not found.**) or Paint Tool SAI, since it is also a type of art program.

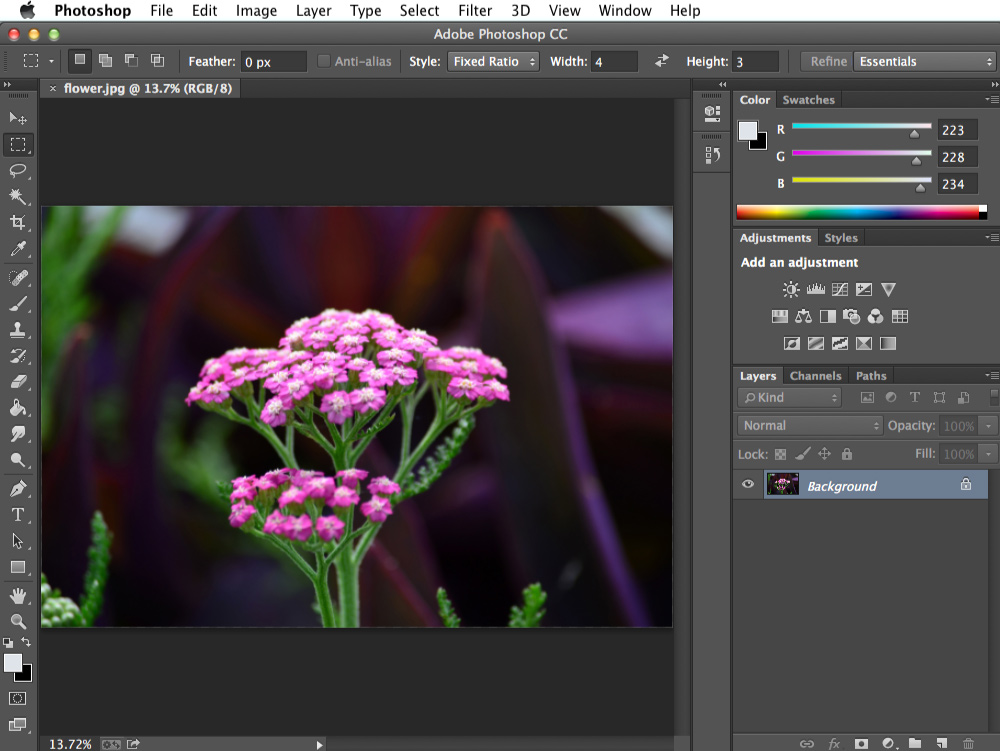


Figure 8 - Photoshop User Interface (GCFGlobal, n.d.)

Another general principle for UI design is to be aesthetically pleasing. This is an important aspect of the project. It can be achieved by using colour and graphics effectively. It also helps to convey messages clearly.

Limiting the ability to make mistakes can greatly improve usability. The UI needs to be forgiving and always have an option to undo an action. Finally, a simple design helps to not confuse new users by not overwhelming them with an overcrowded interface. Limiting the number of things the user has to remember also helps to reduce memory work (Galitz, 2007).

Identifying the user base is also vital for the success of a program (MacDonald, 2008). More influential factors for the project are the users’ familiarity with the computer, frequency of use, gender, and age. The users of a cross stitch software will mostly be discretionary users, so it will only be used if the benefit of using it exceeds the effort learning it, according to Galitz (2007).

Failing to use the principles will make the UI inefficient and people will have difficulty doing their job. They might even stop using the program.

* + 1. Colour

Colour can be really powerful. It can convey emotions without saying anything (Barševska & Rakele, 2019). It is not a surprise that it has a great impact on the mood and behaviours of users. It provides structure and meaning to the screen. Research proved that colour in user interfaces improves performance and aids memory (Galitz, 2007). However, some combinations can strain the eye, so it is important to study colour theory to make the right decisions. There are different ways of selecting a good colour scheme. Based on colour theory, there are the following main combinations (see all of them in Figure 9):

* Monochromatic, which uses one colour with different hues.
* Complementary, which combines contrasting colours (red, green). It is difficult to harmoniously combine them in a UI.
* Split complementary, where a primary colour is used with two analogous colours to its complement.
* Triad combines three colours on the colour wheel with equal distance from each other.

Graphical user interface

Description automatically generated

Graphical user interface

Description automatically generated Graphical user interface

Description automatically generated

Graphical user interface, application, website

Description automatically generated

Figure 9 - Colour combinations (Barševska & Rakele, 2019).

It is recommended to start designing the UI in black and white first. This helps focusing on the layout and simplifies the process. When the design is ready, then start adding colours as needed, but only with meaningful purposes. However, no more than five colours should be used. Avoiding pure black is encouraged since it looks unnatural. Keeping in mind the common meaning of colours will help the design. For example, red usually has meanings like stop and danger, so it shouldn’t be used for an OK button (Barševska & Rakele, 2019).

* 1. Existing Cross Stitch Software

This section provides an overview of existing cross stitch pattern making software. Knowing what is currently available will help inform this project. There are a number of cross stitch pattern making software on the market with varied quality and features. The most popular and advanced one is WinStitch/MacStitch (2019). This is a commercial software, and it is regularly updated. Another example is PC Stitch (2016), which was the preferred software until it stopped getting updates. Both can be quite expensive for a hobbyist, so a good alternative can be free, open-source software. These generally have less features and not as clean UI (user interface); however, they can still create a pattern. These include XStitch (Chestnut Pens, 2020) and CStitch (Klein, 2017).

All of the above software can convert a picture into a pattern. WinStitch has a wide range of features and a professional looking UI, so it is a good source of ideas for this project. Some features proving very useful are the ability to select the size of the pattern before generating it, as well as the number of colours used. These types of features are ones that will need to be considered in the design phase of this project. WinStitch uses real thread colours in the pattern, this makes it very practical for the user to buy supplies. See on **Error! Reference source not found.** that the thread colours are as close as possible to the original painting, and how detailed the pattern is. There are also a lot of options to edit the pattern, from changing thread colours to drawing on it. The biggest differences between the free and commercial applications are the number of features and how the user interface looks. WinStitch’s UI is very sleek and intuitive, while CStitch’s is quite antiquated and sometimes hard to use. One of the aims here is to design an attractive UI with all suitable features without having too many options that may overwhelm the user. The intention for this project is to be a good middle ground for users, available at a lower price while still maintaining quality.



Figure 10 *- Painting and cross stitch. Pattern created in WinStitch (Batho, 2014).*

* 1. Summary

After reviewing and analysing past literature about relevant topics, I am more prepared to design and implement the application. A simple resampling algorithm can be used for the basic implementation and fast performance, however, a more advanced one is preferred to preserve detail. WPF enables developers to build a robust application with advanced graphics and highly customized UI. This is ideal based on the requirements of the project, therefore WPF will be used instead of Windows Forms. The UI needs to be simple, so that novice users can interact with it easily, and optimal colour palettes need to be considered for a harmonious look.

# Project Planning

## Introduction

This chapter considers which methodology the project development will follow and what it is required to do. Furthermore, it presents the tools and techniques which will be used to implement the project alongside an explanation of legal and ethical issues that may occur.   
This chapter explores methodologies for project development and discusses possible tools and programming languages. Furthermore, the requirements are explained. And finally, it investigates legal and ethical issues in software development and considers possible problems in this project.

## Methodology

Spiral? Some agile things.

For a successful outcome it is important to decide the methodology used for the project management. There are a few things to take into consideration before deciding. The developer is only one person in this project so there is no team to be able to help; this means that some techniques from well known methodologies can’t be used that were created for teams, like Scrum. There are limited resources and a limited time frame, so good time management is crucial. Requirements might change along the line, so the approach needs to be flexible enough to allow that.

The waterfall model is a sequential life-cycle where the phases are clearly defined (See Figure 11). The next phase can’t be started if the previous one is not completed. The features are also fixed. It is usually used for bigger projects. The advantage is that all requirements are clear before the start of the development and everything is well documented, however if a problem is discovered after a phase is finished it can’t be easily fixed (Balaji & Murugaiyan ,2012).

Diagram

Description automatically generated

Figure 11 -Waterfall method (Hughey, 2009).

Agile is an iterative development methodology. There is no extensive design upfront and it allows for changes. The pros of Agile is that the client can see what the program is like during development and change it according to their requirements, this ensures their satisfaction at the end of the project. The cons are that it can be more difficult to understand than Waterfall, there can be a lack of documentation, and it can be inefficient for large organizations.

The Agile method’s main characteristic is that it is more flexible than older methodologies like Waterfall (Tordrup Heeager, 2014). This is useful in modern software development, since the features often change and evolve during the project. Based on this aspect it would be a good fit for this project. The high-level requirements have been decided, and the details are explored during development. It is likely that they could change in the future, which Agile can accommodate for. Another important aspect is effective time management, since time is one of the biggest constraints of the project. Timeboxing is an Agile technique that divides the development into iterations. Each iteration is one or two weeks, which fits in perfectly with the schedule of the supervisory meetings. This also allows a step-by-step approach to development, which makes it less overwhelming and divides up the workload to smaller parts. In Agile, testing is done after each iteration to make sure that the new features work together with the product (See Figure 12). This ensures that the software is functional by the time it is finished, unlike in the waterfall method where the testing is done at the end of the life cycle.

Diagram

Description automatically generated

Figure 12 - Iterative Agile methodology (Hazevytch, 2020).

The Kanban method is also useful. A board can be created on Trello.com, which visualises tasks. With the help of this it’s easier to see which features need to be implemented and which were already done. It can also motivate the developer to see how much work has already been done. The tasks can be prioritised and put into different categories.

Communication with customer, like supervisor.

## Requirements

Requirements were gathered during the planning phase of the project. These requirements are based on existing cross stitch software, and on what I would want an application to have as a cross stitcher.

The requirements are prioritized using Moscow. This method sorts features into priority order in an easy to understand way (Waters, 2009). The priorities can change at the end of each timebox based on needs. The most important features are listed first.

### Must

* A Windows application that can create a pixelated downsampled image.
* Basic UI for the application.
* A grid over the pattern.
* The ability to save the pattern.

### Should

• The ability to convert an image into different size of patterns.

• A more advanced UI.

• Saving the pattern as a PDF.

• The ability to edit the pattern once it was generated.

• The ability to reduce the number of colours used in the pattern (quantization).

### Could

• The ability to use real thread colours in the pattern.

• The ability to pan and zoom in on the pattern while editing.

### Won’t

• Using different types of stitches, like back stitches and French knots.

• Representation of how the pattern will look once finished (preview).

• Blended thread colours, which gives more colours.

## Potential Solutions and Tools and Techniques

Desktop application, web application?

Windows, mac, linux?

There are several potential ways of creating a program like this, one of them is developing a desktop application. It would reach a greater user base if it could run on several operating systems, however because of time limitations it will be developed for only Windows since this is the most widely used operating system (Shanhong, 2021). Another option is creating a mobile application; however this wouldn’t be sufficient since enough screen space is needed to be able to edit the pattern.

A web application could be considered too. This way the user doesn’t have to download anything. The main obstacle with that is that it wouldn’t be possible to do in the time frame due to no experience in that field and the steep learning curve. Another downside is that an internet connection would be also required while using the program.

### Application platforms

Wpf, winforms, uwp, win32, winui 3 different ways of doing it, confusing in self reflection. Different pltaforms

There are various platforms to consider, each with their own pros and cons.

The Win32 API is the most performant of all of them. It has direct access to hardware. It uses C or C++, which is the language I have the most familiarity with. It is an unmanaged environment (directly executed by the operating system), unlike .NET. It doesn’t offer an editor, so the UI has to be created in code using Windows functions. It is a great option for applications that need the highest efficiency and low-level control, but this project doesn’t require that. It also has a steeper learning curve than APIs with an inbuilt UI editor (Microsoft Docs., 2020).

UWP (Universal Windows Platform) is the newest API from the ones considered. It works on all platforms as long as it uses windows 10 or newer. XAML is used for the UI and C# or C++ for the code behind. It is suitable for sophisticated UI and graphics-intensive scenarios (Microsoft Docs., 2020). A con is that there aren’t as much documentation available as for WPF or Winforms, since it is relatively new. It would also be important for the application to work on older versions of Windows too, since a lot of the potential users might not have the newest operating system.

WPF and Winforms were discussed in detail in section 2.4. The advantage of both is that they have an inbuilt GUI designer, which speeds up the UI development process, and they support Windows 7 and later versions (Microsoft Docs., 2020). WPF supports high quality 2D graphics, like images, geometry, and shapes. It also supports the use of pixel shaders (version 3.0), which allows for the implementation of effects. Takes advantage of the GPU and minimizes CPU usage. A useful feature of WPF is data binding; this enables a connection between a UI control and the data it displays. If the data changes, the UI reflects that automatically. Winforms doesn’t have these previous features, so WPF is selected as a platform for development.

C#, XAML C++ VB

### Programming languages

The programming language used in the project could be C#, C++ or Visual Basic.NET based on which application platform is chosen.

C++ is a general-purpose programming language designed by Bjarne Stroustrup (2007). He first developed “C with Classes” as an extension of C, and then using his experiences gained from that project created C++. It is a low-level, strictly typed language. It allows direct access to the memory, which makes it very efficient. It is best suited for performance critical programs. A downside could be that the memory has to be managed manually, unlike in the two other languages. I am most familiar with C++ so being able to use it in the project would be an advantage. WPF supports C++ in theory, however, the UI editor can’t be used. This would undermine the main point of using WPF. UWP and Win32 supports C++.

C# was developed by Microsoft as part of .NET in 2002. It is a strictly typed, object-oriented language, higher level than C++ (Microsoft Docs., 2021). It provides automatic memory management called garbage collection. This ensures there are no memory leaks, and programmer don’t have to deal with memory management (Bates, 2004). This also means that memory is harder to access. It can only be accessed inside blocks marked with the unsafe keyword. Memory cannot be explicitly freed, only garbage collection can do it. Once there are no references to an object it is collected. It is similar to C++ but there is still a slight difference, which adds some difficulty to the development. It is supported by WPF and UWP.

Visual Basic.NET is an object-oriented language. It is similar to C# in the sense that they are both higher level managed languages and are built on the .NET framework (Bell, 2002). It also uses automatic garbage collection. It is considered to be easier to learn than C-type languages because it is more similar to normal English. However, because of previous experience in C based languages it wouldn’t necessarily be easier for me, and making a change to VB would just slow down development considerably. WPF supports both C# and VB, they are functionally equivalent.

A good challenge to learn XAML and improve my C# skills.

Not case sensitive. Cannot use pointers.

### Other languages/Data languages

#### XAML

XAML is an XML based markup language developed by Microsoft in 2008. It is used in several .NET technologies, like WPF and Silverlight. It is also used in UWP. In all the mentioned technologies it is used for creating the user interface. Everything can be done in code behind that is done in XAML, however the advantages of using it is that it’s more concise and readable (Microsoft Docs., 2017). The separation of the UI and code behind encourages clean architecture. The UI can be modified without changing the application logic. It is easy to learn, even graphic designers can use it. A disadvantage of it is that it cannot contain logic, so all event handlers need to be declared in code. In WPF the XAML can be automatically generated based on the UI editor, and it can be written manually too (Microsoft Docs., 2020). Case sensitive.

HLSL maybe GLSL

#### HLSL

WPF uses HLSL for its effects. HLSL (High Level Shader Language) is a graphics programming language developed by Microsoft in 2003. It was introduced with DirectX 9 and Shader version 2.0. Before HLSL shaders were developed in a low-level language similar to assembly. Registers had to be managed by the programmer. HLSL was made to simplify and abstract shader development (St-Laurent, 2005). The syntax is similar to C languages but it has its own data types. The advantage of using shaders is that the calculations are done parallelly on the GPU, not using CPU, making the program much faster. Data is sent over from the CPU side and it’s rendered using a pipeline. The main parts of the graphics pipeline are vertex shaders, geometry shaders and pixel shaders (Li & Xu, 2009).

WPF can apply inbuilt effects to a control, like blur or drop shadow. Custom shader can also be written in HLSL. WPF only supports Shader model 3.0 pixel shaders. The final colour of a pixel is decided in the pixel shader. The advantage of using shaders in the program is that it makes the image processing significantly faster. GLSL (OpenGL Shading Language) could also be used for programming shaders, however WPF doesn’t support it.

### Other tools

#### IDE

Visual Studio is an integrated development environment created by Microsoft. It is a code editor that supports programmers in development. It helps reducing mistakes and increases speed. Its features include IntelliSense code completion, code refactoring, integrated debugger, and syntax highlighting for easier coding (Muşlu et al, 2012). Both native and managed code can be written. It supports several languages including C++, C# and VB. The Community edition is free for everyone and the Enterprise versions is accessible for free for students. The .Net framework was built to work with visual studio, so it provides a comprehensive set of tools for development with .Net. The downside of it is that it takes up a lot of memory on the computer. Visual Studio Code could be another option for an IDE. It supports a lot more languages than Visual Studio, however it doesn’t have key features like the UI editor, which is important for the development.

breakpoints to find problems faster.

#### Version control

GitHub is a cloud-based repository hosting service and code sharing platform (Loeliger & McCullough, 2012). It is essential to back up work while developing a project and GitHub helps exactly with that while offering a lot more features too. The code can be managed and tracked using Git based version control. Git was developed by Linus Torvalds. The way it is used is that first a repository is created and then uploaded on the server. This can be accessed from any computer with the log in details. The repository can be public or private. All changes are recorded, and the older versions of the project can be accessed too. If a mistake was made the code can be rolled back to a previous state. Changes can be committed without internet connection. It can be also used for cooperation (Spinellis, 2012). Users can browse its largest collection of open source software (Borges et al, 2016). Feliciano et al (2016) found that using GitHub benefits students in several ways, for example it familiarizes us with a popular industry tool.

GitHub can be used in different ways. A fast way is using it from the command line; however, it can be difficult to remember commands. It can also be linked to Visual Studio, but this proved to be unstable and unreliable at times. The best and easiest way was found to be using the GitHub Desktop application. It is helpful for students because it visualizes the process and features a clean user interface (See Figure 13).

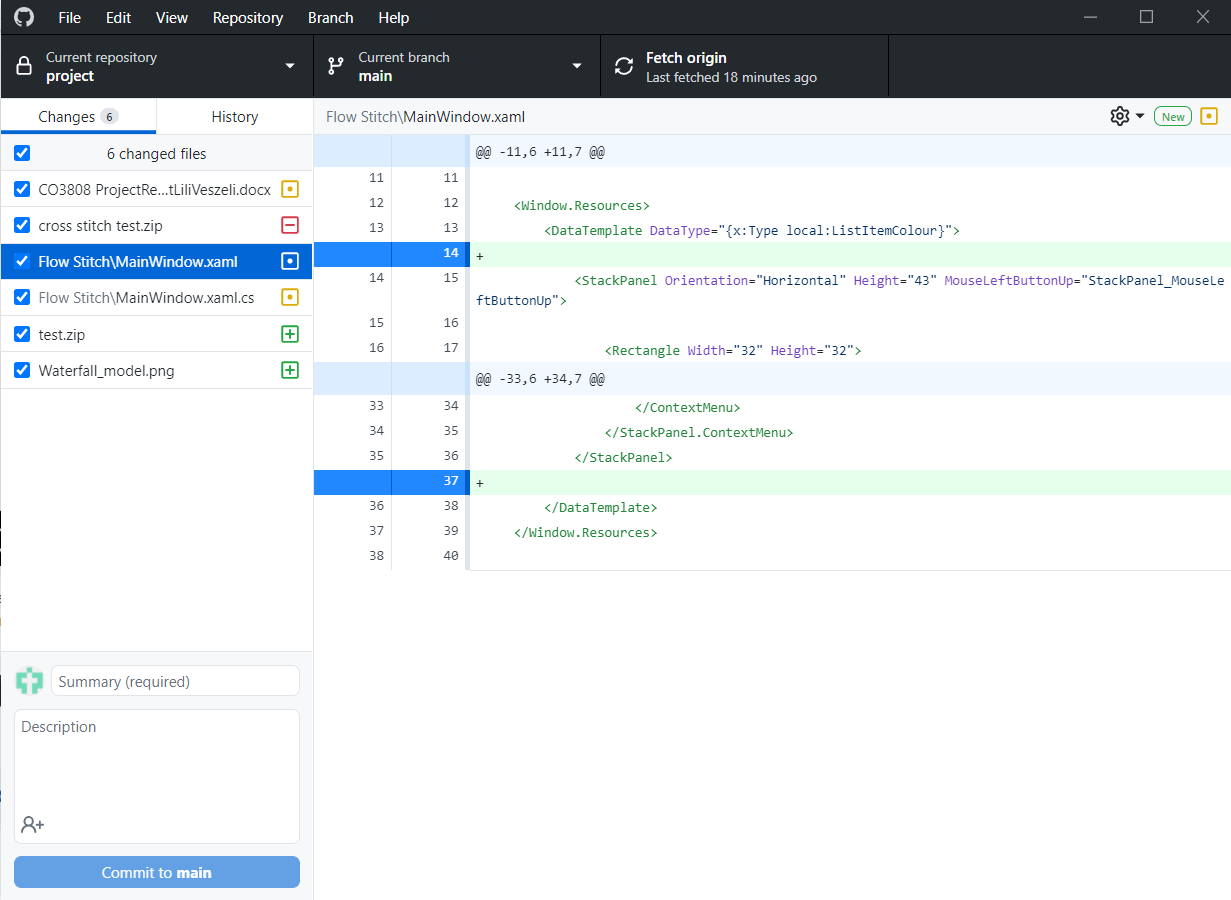


Figure 13 - GitHub Desktop.

## Legal, Social, and Ethical Issues

### Ethical Issues

The role of ethics in software design has increased recently. The issues can be categorized into privacy, accuracy, property, and accessibility (Thomson & Scmoldt, 2001). Improper access to private information causes privacy concerns. This application does not utilize any personal data from the user, so privacy issues do not arise. Accuracy and property issues are also irrelevant.

The issue with accuracy comes up when the program does not work as it was expected to. In theory the developer is bound to fix all problems that the users encounter, especially if it is a commercial product, however sometimes fixing a bug introduces bigger problems. Most of the time significant bugs can be avoided with thorough testing.

Property issues include the knowledge possessed by individuals that was used in the development process. People contribute knowledge and skills to the project, which can lead to issues with intellectual property rights.

To have appropriate access to a program, a user must be able to provide the required input and comprehend the information output according to Thomson & Scmoldt (2001). The lack of the availability of a language in a program that the user understands causes an accessibility issue. The languages should be chosen based on the target audience. If concepts are used beyond the user’s understanding, then that can cause a problem.

By promoting cross stitch a needle could stab the user lol.

### Accessibility

Accessibility for people with special needs is an important issue in today’s society (Kavcic, 2005). It is often taken for granted that a user can read information displayed on the screen, can hear sounds and can use the keyboard and mouse. However, not all of the potential users are capable of these. Supporting alternative input devices and allowing keyboard only access can help people with mobility issues, who can’t operate a mouse. Resizable text and good contrast between colours can support people with visibility problems. Flashing elements should be avoided. For hearing considerations, any audio has to be represented visually, like with subtitles. This project doesn’t have any audio components, so it’s no applicable.

#### Colour blindness

Some colours in the application and pattern might be hard to distinguish for colour blind people. 1 in 12 man has a colour-vision deficiency. Green and red are usually the most problematic, but other colours containing them can be affected too. A solution would be using a colour blind friendly colour palette (Rigden, 1999). However, these palettes can drastically decrease the number of colours available, especially if real thread colours are used. Another solution is using symbols to illustrate colours during editing, this way all colours can be used.

### Legal Issues

A legal issue could be libraries used that need license for distribution. All libraries used are open source (allowing the code to be freely used, modified, and shared), however they still have licenses. AForge has a GNU General Public License, version 3 (GPLv3). This is the most restrictive open source license: the source must be made public, and the software must be released under the same license. Csv helper uses Apache License 2.0; with this license the source code doesn’t have to be public, the software can use any license, and modifications need to be documented. Other libraries use the MIT license, which is the most permissive one: the source doesn’t have to be public, it can use any license, and changes don’t need to be documented. The license needs to be added as a text file to the root directory of the project (Raicea, 2017).

License texts are in Appendix 3 – Software Licenses.

## Summary

In this section, Waterfall and Agile methodologies were considered, proving Agile techniques to be more appropriate. Further on, several APIs were compared resulting in WPF being the most suitable. Additional tools were explored with their advantages and disadvantages. And legal and ethical issues were discussed, highlighting important design considerations for inclusivity.

# Design

## Introduction

This chapter explores a useful design pattern for WPF applications, which helps to make the code cleaner and the testing easier. Furthermore, it outlines the essential use case scenarios for the system design; and discusses the ideas behind the user interface design.

## Software design

Design patterns. Singleton, command.

Solid principle.

Design patterns are a solution to common programming problems (Johnson & Vlissides, 1995). It is like a template that can be applied to different scenarios. An architectural pattern is similar to a design pattern, but it applies to the overall system structure (Avgeriou & Zdun, 2005).

### MVVM Pattern

Model-View-ViewModel (WVVM) is an architectural pattern often used in WPF applications. The pattern was published by John Gossman, an architect of WPF, in 2005 (Sharma, 2019). It’s based on Martin Fowler’s Presentation Model pattern. It is similar to the widely used MVC (Model-View-Controller) pattern in the way that both separate the UI from the presentation logic (Microsoft Docs., 2012). One of the advantages of using this pattern is that it prevents tight coupling between the UI controls and the code behind. Tight coupling can cause difficulties when trying to update the user interface. The whole UI can be redesigned without any changes in the code if using MVVM. It is also easier to test the program using unit testing. Furthermore, it enables data binding, which is a powerful feature for displaying data in WPF. The code is also cleaner and easier to understand for future developers. A disadvantage of the pattern is that debugging is difficult when using data binding. Additionally, using the pattern for a simple UI can be excessive, and for a complex UI it could prove difficult to design.

There are three key components in the pattern: model, view, and view model. Figure 14 shows the relationship between them.

Helps designers and programmers work together. Loose coupling.

Graphical user interface, application

Description automatically generated

Figure 14 - Model-View-ViewModel components.

The view represents what can be seen on the screen, the user interface in this case. It is defined only with XAML in WPF. It should not contain any business logic, but it can handle some basic user interaction. The view model is connected to it by databinding.

The model is the data of the application. It could be phone numbers, addresses, etc. The business logic is separate from the model.

The view model is an intermediary between the two. The user can modify the with the help of the view model. It retrieves the data from the model and sends it to the view in an easily usable form. It also implements commands that the user can invoke from the view. In order for this two-way databinding to work, the INotifyPropertyChanged interface has to be implemented. This raises the PropertyChanged event, which updates the data in the view (Microsoft Docs., 2012).

In this project the model is the list of thread colours. The view is the palette displayed in the UI. The two are connected with data binding. The model view does the updates with the help of the INotifyPropertyChanged interface.

Software refactoring?

## System Design

To be able to implement the application it is essential to know the different actions and use cases that the user would want to do.

### Pattern Creation

* First, the user will be able to open a new picture. The height of the pattern and the maximum number of thread colours will have to be input. Then the picture will be chosen using the Windows file dialog.
* Then the chosen image will be downsampled and quantized.
* The quantized colours will be converted to the closest real thread colours. These colours will be displayed in a palette next to the pattern.
* The pattern will be able to be saved as a high resolution image.
* The pattern will be able to be saved with symbols that represent the different colours.

### Pattern Editing

* The user will be able to edit the output pattern by clicking on individual stitches. The stitches will change their colour to the current drawing colour if clicked (initially black).
* The user will also be able to erase stitches by clicking on the eraser button and then clicking on one of the stitches. This will make them white.
* A new drawing colour will be able to be chosen when clicking on the colour wheel or the pencil button. If the colour wheel is clicked, then an RGB colour picker window will open up and the desired colour will be able to be selected. This will be then converted to a thread colour. If the pencil button is clicked, then a list of the thread colours will open in a new window and one can be chosen.
* The colours in the palette will be able to be selected too to draw with by clicking on them.
* The palette colours will be able to be changed by right clicking them and selecting the corresponding option from the context menu. The new colour will be selected with the colour picker, converted to DMC colour, and all the stitches will be changed to it.
* Colours will be able to be deleted from the palette too with the context menu. This will delete it from the palette list and requantizes the pattern using the reduced number of colours.
* The pattern will be able to be previewed. This will show how a finished cross stitch would look like based on the pattern.

### User Interface

* The undo/redo functionality will be available. This will restore the previous state of the pattern. It will be able to be accessed from the top menu or by pressing the Ctrl+Z and Ctrl+Y key combinations.
* The user will be able to pan and zoom while editing the pattern. This is especially important for bigger patterns.
* Information about the application will be able to be found by pressing the About button in the top menu.

## User Interface Design

Existing program ui, compare. Common expectation.

The aim of designing the user interface for the program is to make it as clean and intuitive as possible. It is also important to be aesthetically pleasing. The design is based on other popular creative applications and existing cross stitch programs. The familiarity with these existing programs gives an idea to the user how to use this program too. The first storyboard was largely based on Paint Tool SAI’s user interface (See Figure 15), and the idea for the way to display the thread colours was from WinStitch (See Figure 16). The storyboard and the final UI design can be seen on Figure 17 and Figure 18.

Graphical user interface, application, Word

Description automatically generated

Figure 15 - Paint Tool Sai UI.

A picture containing text, tape, electronics, display

Description automatically generated

Figure 16 - WinStitch UI.

A drawing on a white board

Description automatically generated with low confidence

Figure 17 - Initial Flow Stitch storyboard.

Graphical user interface

Description automatically generated

Figure 18 - Flow Stitch final UI.

Flow Stitch’s user interface is much less busy than the programs it was inspired by; this is because it has less features. However, this also makes it easier to navigate for new users while still being a functional pattern maker. The menu bar is quite standard, it works the same as Word or other drawing programs: a new picture can be opened and saved, and the application can be exited. Under the Edit item the buttons for undo/redo functionality can be found. This allows for the user to make mistakes and restore the pattern. It can be also used with key commands. Finally, under the About part in the menu, some information can be found about the program. Then there is a side panel containing all the editing tools and data about the pattern. The palette resembles the one used in WinStitch, however, it is a bit bigger for better visibility. The organization of the drawing tools are similar to how it is done in SAI. The main drawing area has the pattern in its pixelized and quantized form. This can be zoomed in and panned.

According to the original storyboard, a colour wheel would have been in the side panel too. This had to be taken out of the main UI since WPF didn’t have a control for it, and no libraries offered that functionality. Instead, the colour picker opens in a new window, which was achieved by the WpfColorPicker library made by Dsafa on GitHub.

Some changes were made during development based on my supervisor’s recommendations. For example, the pattern was centred instead of being on the right, and the palette was made wider to allow for the full colour name to be read. Features were also added, like the properties plane to provide more information about the pattern, and a splash screen that shows as the program starts up.

After user testing the participants said it was easy to understand which UI component did what. They suggested some improvements for

## Summary

The advantages and disadvantages of the MVVM pattern were discussed and how it can apply to the project. It is useful for the colour palette; however, it won’t be implemented for the full application because the rest of the interface is quite simple. In system design important features and use cases were outlined in detail, which will help the implementation go smoother. The decisions behind the user interface design were reviewed; and the changes along the development were demonstrated.

# Implementation

## Introduction

Each of your chapters should have an introduction to tell your readers what they will find in the chapter. Depending on what type of project you are doing, you may name this chapter ‘Investigation’.

## Pattern creation

### Downsampling

Bicubic, using gdi library. Can input desired height of pattern.

Issue with black line, was fixed. Needed to wrap image.

### Colour quantization

Aforge , Wu quantizer , others, Implement myself, then exactly know how it does it

Chosen because it has a function that accepts the number of colours to what it reduces the palette.

It is also able to make an image use a certain palette.

Not the most precise. Sometimes returns less colours than the input value, especially if it is a bigger number. However, it is not always possible to get that exact number of colours, it depends on the image.

### DMC Thread colours

Dollfus-Mieg et Compagnie is a French textile company created in 1746 (DMC, n.d.). They make one of the most widely used embroidery threads in the world. It has the largest colour range (482) compared to other threads like Anchor; and it’s made from Egyptian cotton (Bare, 2018).

These thread colours are used in the program. The RGB value, name, and number for each thread is stored in a csv file. These are read into a list at the start of the program using the CsvHelper library. A DMC class was created specifically for these colours (See Listing 1). The list is storing this DMC type.

//class to store DMC thread colour properties

public class DMC

{

public string Floss { get; set; } //identifying number of colour

public string Description { get; set; } //name of colour

//RGB values of colour

public int Red { get; set; }

public int Green { get; set; }

public int Blue { get; set; }

}

Listing 1 – [DMCColours.cs] DMC class.

DMC (n.d.). ABOUT US. Retrieved April 28, 2021 from <https://www.dmc.com/us/p-us-about-us.html>.

Bare, T. (2018). 5 Cross Stitch Floss Brands Compared. Retrieved April 28, 2021, from https://www.thread-bare.com/2018/10/16/5-cross-stitch-floss-brands-compared

The final quantized colours of the image are converted into real life DMC thread colours. The colours are first converted to Lab colour space, since that is a better representation of how humans perceive colour, so better chance to get a good thread colour match. The Lab colour space is represented in three dimensions, as illustrated on Figure 19. There is one dimension for lightness (*L*, where *L*=0 is white and L=100 is black), one for indicating the value between red and green (*a*, where the lower *a* is the greener the colour), and finally one that shows thee value between yellow and blue (*b*, where the lower the value the bluer the colour). It includes all perceivable colours by the human eye, which means that is exceeds the RGB colour space (See Figure 20) (Rathorel et al, 2012).

Diagram

Description automatically generatedChart, radar chart

Description automatically generated

Figure 19 – Modell of the Lab colour space. Figure 20 - Colour coverage of Lab.

Rathorel, V. S. ,M., Kumar, S. M., & Verma, A. (2012). *International Journal of Emerging Technology and Advanced Engineering.* Colour Based Image Segmentation Using L\*A\*B\* Colour Space Based On Genetic Algorithm (Vol. 2, Issue 6). www.ijetae.com

Blue value had to be tweaked using a heuristic.

//getting a closer blue colour because of lack of bright blue threads

var b = myPalette.Colors[i].B;

if (myPalette.Colors[i].B > 180 && myPalette.Colors[i].G < 100 && myPalette.Colors[i].R < 100)

{

b = (byte)(160 + (20 \* (b - 160)) / (256 - 160));

}

Listing 2 - Heuristic used for more accurate colour matching.

Introduce Lab colours.

Find another alternative for colormine, or implement for myself.

Colormine library searches for closest colours. Smallest delta E.

Colour palette created on the side.

## Pattern editing

Click on pixels, changes to current colour.

Colour picker (Dsafa). Choose rgb colour that is automatically converted to DMC. Or choose dmc colour. Can change a specific colour. Can delete a colour. Can erase stitches.

Palette updated.

## Pattern symbols

Small symbols added on top of stitches, to differentiate them easier. Each colour has different symbol. Drawing image, drawing group.

## Preview

Show how the finished cross stitch would look, using a picture of a cross stitch on aida canvas.

### Colour blending

Multiplicative blending. Getpixel, setpixel. Pointers. Slow.

### Shaders in WPF

Fast. Multiply original image with the passed colour.

Shader wrapper. In .fx file. Need to compile to make .ps file that is actually used. Need to set the effect to a control.

## User Interface

Palette, buttons (draw, colour picker, eraser, symbol, preview). Menu, undo redo, new, save, about.

Put in design.

### Undo/Redo functionality

Image stored in a list.

Explain how the pattern is created?

The libraries used?

* Aforge - LGPL v3(GNU Lesser General Public License) – used as DLL and not as static library.
* Colormine – open source. Why lab colours. Rgb hsl
* Dsafa – MIT license
* CsvHelper – open source (Microsoft Public License (MS-PL), Apache License, Version 2.0)
* GDI+ API

Class diagram? No

Code can be formatted using the ‘Code’ style. An example is shown below. It can be a little bit tricky to keep the formatting when pasting from an IDE but the following works for most IDEs: Copy the text from the IDE, paste it in Word, select the pasted code and change the style to ‘code’. It is worth noting that spell checking is deactivated for the ‘Code’ style.

Captions are entered through the ribbon menu under ‘References’ -> ‘Insert Caption’. Select ‘Listing’ (or add a new Label called ‘Listing’ if it does not already exist) and add the caption text in the white box, separated with a dash as the example above shows. Think about a naming convention for listings and stick to it throughout the report. For example, as seen above, ‘[ClassName or Filename] Description’.

In case you are mixing multiple programming languages: Consider stating the language name in the caption if it is not obvious from the file name or when there is no file name to refer to. For example, when you use XML and HTML, JavaScript and TypeScript or other languages with similar syntax. A suggestion might be to add the language in parenthesis at the end.

It is also possible to use the ‘code’ style “inline” to highlight commands in normal text by selecting the words to highlight and choosing the ‘code’ style. For example: This example demonstrates the ping 127.0.0.1 command.

## Summary

Write a short summary at the end of each chapter. Do not use the words ‘In summary’, we know what it is from the title.

# Testing

## Introduction

Each of your chapters should have an introduction to tell your readers what they will find in the chapter. In this chapter you should introduce your test strategy – how have you tested your artefact. You should also talk about user testing. How did you test with real people? How did you select them? What did you ask them to do? What ethical considerations did you adhere to? In this chapter you will also discuss how you have carried out an evaluation of your artefact. This is not the same thing as a total project evaluation.

This chapter introduces key testing strategies and describes how the application was tested.

About testing and how the app was tested.

Importance and aim of testing.

Talking about different types of testing: white box, black box, unit tests

Automation testing

Cat stripe testing

Singh, S. K., & Singh, A. (2012). Software testing. Vandana Publications.

Singh, Y. (2011). Software Testing. Cambridge: Cambridge University Press. doi:10.1017/CBO9781139196185

Mili, A., & Tchier, F. (2015). *Software testing : Concepts and operations*. ProQuest Ebook Central [https://ebookcentral.proquest.com](https://ebookcentral.proquest.com/)

## The Importance of Testing / General?

Nowadays many different fields depend on software. The expectations are much higher since the quality of the software can be critical in many real life scenarios and the financial stakes can be high if it fails (Mili & Tchier, 2015). Software testing can assure software quality. The aim of testing is to make sure that the program does what it is intended to do. Testing also increases user confidence, ensures customer satisfaction, and lowers maintenance cost. There are different strategies to testing, each one tests different aspects of the program. They can be divided into functional (unit, integration, acceptance), and non-functional (security, performance, usability) test. The way testing works generally is, the software is executed with desired inputs, and the outputs are compared with the expected ones. If the two don’t match, then there is a bug. A good test case is one that has a high probability of finding errors (Singh, 2012).

## Unit Testing

Unit testing tests the individual parts of the software, it is done at the lowest level. It is usually performed by the developers. Testing boundaries is often done in unit testing because it has a higher probability of detecting faults. It can also be automated, which speeds up the process and can cover more code areas; a good tool for that is NUnit. However, it has limitations: it is impossible to evaluate every execution path, and there’s a limit to how many test scenarios can be made. Furthermore, parts of the software working independently of each other doesn’t mean that the whole product is functional. Integration testing is needed too, which is testing two or more units together (Singh, 2011).

To add a caption to a table, either select the whole table (e.g. by clicking on the + symbol in the upper left corner of the table), right-click it and choose ‘Insert Caption’ or click in any table cell and select ‘References’ -> ‘Insert Caption’ from the ribbon menu. Choose ‘Table’ as label and ‘above the item’ as position. Add the caption text in the box, separated with a dash as the example below shows.

Table 1 - Test Results

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Non-Functional Testing

Usability testing and user interface testing.

## User Testing

This testing method involves real users interacting with the application.

## Automation Testing

Automation vs manual

It generates the test data automatically for the software. It’s more efficient, cost-effective and reliable if done correctly, however, it is not a straightforward process. Test data can be generated by static evaluation (like syntax and semantic analysis tools) or dynamically by executing the program (for example, performance testing or functional testing). Any functional testing can be automated. A test automation pyramid is often used in agile projects (See Figure 19).

Diagram

Description automatically generated

Figure 21 - Ideal test automation pyramid (Palamarchuk, 2015).

The testing could go on forever, so criteria need to be set for when it is considered adequate. This can be different based on the program, but an example is “Every path of the code should be executed at least once”. This sets the standard to measure the thoroughness of testing.

The advantages of automation testing is that it is faster than manual testing, has a wider coverage of the code, it is reliable, consistent, more accurate, and saves time and cost. The disadvantages is that it can be expensive and manual testing is still needed sometimes because it might miss insights that a human could draw (Singh, 2011).

Automation testing could be used on the project, especially automated unit testing. The application might need to be refactored to fit the tools better. Automation testing is more suited to large projects and where the tests need to be repeated a lot. This is a relatively small program so fully automating the testing most likely wouldn’t be worth it because of time constraints and the expense of some tools. It is possible to test it thoroughly manually.

## Evaluation

What I did to test it?

Debugging, what I fixed?

What else could have been done?

Had to call on the Austrian task force to test.

Improvements:

* Prompt user to open new image xd
* add RGB/HEX/HSV colours to the palette too
* highlighting selected tool
* scroll down when new colour is added to the palette
* warning instead of some error messages
* colour picker somewhere else since it is different from pencil/eraser
* tooltip for tools too
* use version numbers

Additional features

* highlight certain coloured stitches

Bugs discovered

## Summary

Write a short summary at the end of each chapter. Do not use the words ‘In summary’, we know what it is from the title.

# Evaluation, Conclusions and Future Work

## Project Objectives

Summarise what you have achieved.

The objective of this project was to produce a program that can generate a cross-stitch pattern based on an image, which can be edited and saved. The aim of the program was to be easy to use for hobbyists and to be aesthetically pleasing.

The technical objectives were reached and even exceeded. The design objectives can be improved.

## Self-Evaluation

This section is about yourself. Be honest. Look at where you were situated at the beginning of the project and where you are now. What have you learnt on a personal level, what have you found out about yourself? Try to reflect upon individual goals, experiences, and incidents. No one is perfect, and it is very likely that you will recall both good and bad experiences.

The purpose of the evaluation process is to highlight strengths, correct performance weaknesses, and develop unused skills and abilities. To do this, you must be willing to recognise areas that need improvement or development.

The experience of delivering a longer project with time-constraints was very valuable both for developing my technical skills and improving as person. Since I haven’t done anything like this before it pushed me in a lot of ways. I think one of the main take away from this experience is that I need more confidence in myself. It was really difficult to start to do things alone without knowing if I am doing them right, but when I managed to start the application, I realized it is not as hard as I expected. I felt relieved and was able to achieve and even exceed my goals. I couldn’t have done this without the help of my project supervisor.

I became much better at individual work and research. I learnt to understand code that was not written by my lecturers and modified it to be able to be used in my application. My debugging skills and my understanding of shaders are way better, which proved useful in other modules too. I was wary of C# before, but now I am more confident in using it, even wish some features were available in C++.

I achieved much more than I expected. I learnt WPF and XAML alone. And was able to do things I would have never imagined without help.

And need to put less pressure on myself. My perfectionistic standards need to be lowered – already have been working on this. Need to believe that I can do something and not give up if I can’t do it or can’t understand it immediately. . Because of my perfectionism I sometimes kept procrastinating doing the project, because if I don’t do it then I can’t fail. I became so scared of it and was paralyzed by this fear, which spiralled into thinking I will never achieve anything in life and will never get a job. Things seemed dark

## Project Evaluation

Stand back and evaluate what you have achieved and how well you have met the objectives. Evaluate your achievements against your objectives in section 3.2. Demonstrate that you have tackled the project in a professional manner.

(The previous paragraph demonstrates the use of automatic cross-references: The ‘3.2’ is a Cross-reference to the text in a numbered item of the document, it is not literal text but a field. The number that appears here will change automatically if the number on the referred-to section is altered, for example if a chapter or section is added or deleted before it. Cross-references are entered using Word's Insert or References menu. Cross-references are set to update automatically when printed but may not do so on-screen beforehand; you can update a field manually on-screen by right-clicking on it and selecting Update field from the pop-up menu or by selecting the whole document and pressing F9.)

I exceeded the objectives of the project based on the Moscow requirements in my technical plan. I implemented all must have features except the grid, which was deemed unnecessary because the stitches were easy to see without it. All the should haves have been implemented, except saving it as a PDF, which was proven to be low priority and maybe unnecessary, since it can be saved as a picture. All the could haves have been implemented. Even the preview feature from the won’t have list was implemented which was a surprise for me, since I didn’t expect to get that far. Especially because the shader was quite involved.

In terms of the UI design I feel like I couldn’t quite achieve what I expected to. WPF UI design turned out to be more difficult than expected. Making a basic UI is easy enough, but making it look beautiful is very very hard. I think I managed to make the UI simple and easy to use. The colours are also aesthetic. I decided on a simple pastel design, since most cross-stitch patterns have a similar mood so it would suit them well.

Wpf what I think? Good? Change anything. How well it worked.

All in all, using WPF proved to be a good decision. All the desired features were able to be implemented. The start was difficult, since there is not just a single right way to do things in WPF. A lot of decisions had to be made at the start, which delayed the development. However, once they were made, the rest of the development process was much quicker. I found plenty of resources on Stack Overflow, which wouldn’t have been possible with UWP for example. The UI could also be created quickly without too much difficulty. A problem was encountered with memory management, which could have been avoided if there was more control over the memory, like in Win32, however it was solved? (See 5.5.2 Shaders in WPF).

If I would do it again, I would research WPF even more before starting. I would implement the MVVM pattern fully and do more extensive testing. To have more control over the memory and UI, and have faster image processing, a lower level API could be used like DirectX. However, this would only be possible if the time frame was longer, since it requires more work to add the same features an in a higher level API.

## Applicability of Findings to the Commercial World

Summarise what you have achieved and how it can apply to the commercial world.

The finished application can be used by cross stitchers. It is a great alternative to expensive programs and provides a good range of features. An everyday user can easily make a usable and nice pattern.

It could be sold commercially, maybe with more added features. Alternatively, if it is released as a free open-source program, then parts of it could be reused and could help others who are trying to make similar applications.

the libraries used allow that. (Most of the libraries are open source and Aforge - LGPL v3(GNU Lesser General Public License) – used as DLL and not as static library).

## Conclusions

Summarise what you have achieved. Do not use the words ‘In conclusion’ or ‘to conclude’ or any derivative of those. We know this is the conclusions from the title.

Satisfied with the project. All features were added that were expected to be added in this timeframe. However, there is always room for improvement. Acquired a good base understanding of WPF, which adds to my skillset. Experienced working on a longer project, which helps with expectations about future development projects.

## Future Work

Explain any limitations in your results and how things might be improved. Discuss how your work might be developed further. Reflect on your results in isolation and in relation to what others have achieved in the same field. This self-analysis is particularly important. You should give a critical evaluation of what went well, and what might be improved.

The application is a functional pattern generator, however compared to the leading programs there is possibility for a lot of improvement.

### Improvement of existing features

Some of the already implemented features could be more polished. For example, when the pattern is first generated, the exact number of colours might differ from the input value. This is due to the colour quantization library used. However, colours can be added or removed afterwards.

The colours also sometimes appear to be duller than expected. This could be because of the conversion of RGB colours to the DMC thread colours. Now the RGB colours are converted to Lab space and then matched to the closest DMC colour. This is the best way for colour matching; however, another algorithm could make the process more sophisticated and choose colours closer to what a person would match them to.

The preview takes longer to load for bigger patterns. For example, a pattern with > 400 height would take a few minutes to generate. This would be dramatically reduced by using shaders for big patterns too (which is already used for smaller ones), however it takes up too much memory and the system runs out of GDI objects.

The downsampling of the image could be improved. Based on the paper written by Gertsner et al. (2012), a smart detail preserving algorithm can be used to downsample images. This could be especially useful for cross stitch patterns, since the details are the most important parts that make the image recognizable, and the backgrounds are usually plain and unimportant.

### Additional Features

As for additional features, based on existing top applications (WinSticth), colour blending could be added. This generates more colours by blending existing thread colours together, resulting in a wider variety and closer colour matches to the original picture. At the moment only the simple cross stitch type is supported. This is the case for most free/smaller programs, however more stitch types could be added in the future, like French knots and backstitches.

An additional useful feature could be printing the DMC colour list together with the pattern. The list containing the colours would have to be exported together with the pattern when saving it. An option to save work in progress could be added. This way the user can continue to work on a pattern later even after closing the application. Similarly to other applications, more brands of threads could be added, like Anchor, and the user could choose which type they want to use.

Can’t use transparent images – transparent part will be black. I guess it can be edited afterwards.

# References

Waters, K. (2009). Prioritization using moscow. Agile Planning, 12, 31.

Shanhong, L. (2021). Market share held by the leading computer (desktop/tablet/console) operating systems worldwide from January 2012 to February 2021. Retrieved April 15 2021 from: https://www.statista.com/statistics/268237/global-market-share-held-by-operating-systems-since-2009/.

Tordrup Heeager, L. (2014). How Can Agile and Documentation-Driven Methods be Meshed in Practice? Retrieved April 15 2021 from: https://link.springer.com/content/pdf/10.1007%2F978-3-319-06862-6.pdf.

Balaji, S., & Murugaiyan, M. S. (2012). Waterfall vs. V-Model vs. Agile: A comparative study on SDLC. International Journal of Information Technology and Business Management, 2(1), 26-30.

Hazevytch, L. (2020) Agile Methodology. [online image]. Devcom. https://devcom.com/tech-blog/agile-advantages-for-business/.

*A Tour of C# - C# Guide | Microsoft Docs*. (2021). Retrieved April 16, 2021, from https://docs.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/

Stroustrup, B. (2007). Evolving a language in and for the real world: C++ 1991-2006. In Proceedings of the third ACM SIGPLAN conference on History of programming languages (pp. 4-1).

Bates, B. (2004). C# as a first language: a comparison with C++. Journal of Computing Sciences in Colleges, 19(3), 89-95.

Bell, D. (2002). Visual basic. Net as a first language: an evaluation. ACM Sigcse Bulletin, 34(4), 107-108.

XAML overview - WPF .NET | Microsoft Docs. (2020). Retrieved April 18, 2021, from https://webcache.googleusercontent.com/search?q=cache:h0oqkNyXIe4J:https://docs.microsoft.com/en-us/dotnet/desktop/wpf/fundamentals/xaml+&cd=1&hl=en&ct=clnk&gl=uk

Xamarin.Forms XAML Basics - Xamarin | Microsoft Docs. (2017). Retrieved April 18, 2021, from https://docs.microsoft.com/en-us/xamarin/xamarin-forms/xaml/xaml-basics/

Li, X., & Xu, M. (2009). Water simulation based on HLSL. In 2009 IEEE International Conference on Network Infrastructure and Digital Content (pp. 1066-1069). IEEE.

St-Laurent, S. (2005). The complete effect and HLSL guide. Paradoxal press.

Muşlu, K., Brun, Y., Holmes, R., Ernst, M. D., & Notkin, D. (2012). Speculative analysis of integrated development environment recommendations. ACM SIGPLAN Notices, 47(10), 669-682.

Loeliger, J., & McCullough, M. (2012). Version Control with Git: Powerful tools and techniques for collaborative software development. " O'Reilly Media, Inc.".

Spinellis, D. (2012). Git. IEEE software, 29(3), 100-101.

Borges, H., Hora, A., & Valente, M. T. (2016). Understanding the factors that impact the popularity of GitHub repositories. In 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME) (pp. 334-344). IEEE.

Feliciano, J., Storey, M. A., & Zagalsky, A. (2016, May). Student experiences using GitHub in software engineering courses: a case study. In 2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C) (pp. 422-431). IEEE.

Thomson, A. J., & Schmoldt, D. L. (2001). Ethics in computer software design and development. Computers and Electronics in Agriculture, 30(1-3), 85-102.

Kavcic, A. (2005). Software accessibility: Recommendations and guidelines. In EUROCON 2005-The International Conference on" Computer as a Tool" (Vol. 2, pp. 1024-1027). IEEE.

Rigden, C. (1999). 'The Eye of the Beholder'-Designing for Colour-Blind Users. British Telecommunications Engineering, 17, 291-295.

Raicea, R. (2017). How open source licenses work and how to add them to your projects. Retrieved April 20, 2021, from: apps.quanticfoundry.com/surveys/answer/gamerprofile/.

Johnson, R., & Vlissides, J. (1995). Design patterns. Elements of Reusable Object-Oriented Software Addison-Wesley, Reading.

Avgeriou, P., & Zdun, U. (2005). Architectural patterns revisited-a pattern language.

Palamarchuk, S. (2015). Best Testing Practices for Agile Teams: The Automation Pyramid. Retrieved April 22, 2021 from <https://abstracta.us/blog/test-automation/best-testing-practices-agile-teams-automation-pyramid/>.

Hughey, D. (2009). The Waterfall Development Methodology. Retrieved from: https://www.umsl.edu/~hugheyd/is6840/waterfall.html.

Sharma, N. (2019). MVVM Design Pattern for building robust and scalable iOS apps. Retrieved April 20, 2021 from https://medium.com/codewave/mvvm-design-pattern-c5d9f4a10758.

Microsoft Docs. (2012). The MVVM Pattern | Microsoft Docs. Retrieved April 20, 2021 from https://docs.microsoft.com/en-us/previous-versions/msp-n-p/hh848246(v=pandp.10)?redirectedfrom=MSDN

Afifi, M. (2018). Image Posterization Using Fuzzy Logic and Bilateral Filter. *arXiv preprint arXiv:1802.01009*.

Atkinson, K. R., & Roberts, J. C. (1999). Graphics and Visualization within Cross-Stitch. In Eurographics UK, Conference Proceedings, 1999 (pp. 129-141). Eurographics.

Barševska, Z., & Rakele, O. (2019). COLOR IN UI DESIGN. *PROCEEDINGS OF THE 61st INTERNATIONAL SCIENTIFIC CONFERENCE OF DAUGAVPILS UNIVERSITY*, 79.

Batho, D. (2014). Pieta [painting and cross stitch]. <https://peacockandfig.com/2015/02/from-a-painting-to-a-cross-stitch/>.

Biedl, T. C., Horton, J. D., & López-Ortiz, A. (2005). Cross-Stitching Using Little Thread. In *CCCG* (pp. 199-202).

Chestnut Pens (2020) Cross Stitch Chart Creation Program. Retrieved from: http://www.chestnutpens.co.uk/misc/index.html.

*Choose your Windows app platform - Windows applications | Microsoft Docs*. (2019). Retrieved November 24, 2020, from <https://docs.microsoft.com/en-us/windows/apps/desktop/choose-your-platform>.

Dyer, K. (1997). Counted Cross Stitch Tutorial. Retrieved November 18, 2020, from: [https://www.crosstitch.com/nfaq/nfaq1.html#](https://www.crosstitch.com/nfaq/nfaq1.html).

Fitzgerald, S. (2017). A beginner’s guide to cross stitch – Red Gate Stitchery. Retrieved November 18, 2020, from https://redgatestitchery.com/blogs/red-gate-stitchery/a-step-by-step-guide-to-cross-stitch.

Galitz, W. O. (2007). *The essential guide to user interface design: an introduction to GUI design principles and techniques*. John Wiley & Sons.

Gerstner, T., DeCarlo, D., Alexa, M., Finkelstein, A., Gingold, Y. I., & Nealen, A. (2012). Pixelated image abstraction. In *Expressive* (pp. 29-36).

GFCGlobal (n.d.) Photoshop User Interface [Online image]. GCFGlobal. https://edu.gcfglobal.org/en/photoshopbasics/getting-to-know-the-photoshop-interface/1/.

Griffiths, I., & Adams, M. (2003). *NET Windows forms in a nutshell*. " O'Reilly Media, Inc.".

Gunderloy, M. (2006). *Developer to designer: GUI design for the busy developer*. John Wiley & Sons.

Hohmann, J. (2020) “We are all mad here”: A Content Analysis of Facebook Posts and Comments about the Impact of the Fibre Arts on Mental Health and Wellness.

*Introduction to WPF | Microsoft Docs*. (2016). Retrieved November 26, 2020, from https://docs.microsoft.com/en-gb/dotnet/desktop/wpf/introduction-to-wpf?view=netframeworkdesktop-4.8.

Klein, T. (2017) Cstitch. Retrieved from: http://cstitch.sourceforge.net/index.shtml.

Kwon, O. Y., & Chien, S. I. (2011). Improved posterized color images based on color quantization and contrast enhancement. In *Proc. Int'l Conf. Machine Vision, Image Processing, and Pattern Analysis* (Vol. 5, pp. 1203-1206).

Leslie, C. A. (2007). Needlework through history: An encyclopedia. Greenwood Publishing Group.

MacDonald, M. (2008). *User Interfaces in C#: Windows Forms and Custom Controls*. Apress.

MacStitch and WinStitch. (2019). Retrieved November 18, 2020, from <https://www.ursasoftware.com/macstitch/index.html>.

Mintel (2020). Hobbies and Interest – UK – February 2020. Retrieved from Mintel Reports database.

Misra, A. (2016). Use of Windows Presentation Foundation and Windows Forms in Windows Application Programming. *International Journal of Advanced Research in Computer Science*, *7*(7).

Nathan, A. (2010). *WPF 4 unleashed*. Pearson Education.

Orchard, M. T., & Bouman, C. A. (1991). Color quantization of images. *IEEE transactions on signal processing*, *39*(12), 2677-2690.

*Overview - Windows Forms .NET Framework | Microsoft Docs*. (2017). Retrieved November 25, 2020, from <https://docs.microsoft.com/en-us/dotnet/desktop/winforms/windows-forms-overview?view=netframeworkdesktop-4.8> .

*Overview of .NET Framework | Microsoft Docs.* (2020). Retrieved November 23, 2020, from https://docs.microsoft.com/en-gb/dotnet/framework/get-started/overview.

Parker, J. A., Kenyon, R. V., & Troxel, D. E. (1983). Comparison of interpolating methods for image resampling. *IEEE Transactions on medical imaging*, *2*(1), 31-39.

Parthipan, V. (2017). Image Down-Scaler Using the Box Filter Algorithm.

Patil, M. (2018). Interpolation Techniques in Image Resampling. International Journal of Engineering and Technology. 7. 567-570. 10.14419/ijet.v7i3.34.19383.

PCStitch: Premier Cross Stitch Software. (2016). Retrieved November 19, 2020, from http://www.pcstitch.com/PCS10/PCStitch10.aspx.

RTO (n.d.). RTO Iris Flower Cross Stitch Kit-4-inch x 4inch, 14 Count [Online image]. Amazon. <https://www.amazon.co.uk/RTO-Flower-Stitch-Kit-4-inch-4-inch/dp/B00EW4NCEG>.

Sells, C., & Weinhardt, M. (2006). *Windows Forms 2.0 Programming (Microsoft Net Development Series)*. Addison-Wesley Professional.

Setiabudi, D., Isa, S. M., & Iswanto, B. H. (2017). Digital color classification for colorful cross stitch threads using RGB+Euclidean Distance and LAB+CIE94. Proceedings of 2016 International Conference on Information and Communication Technology and Systems, ICTS 2016, 150–156. <https://doi.org/10.1109/ICTS.2016.7910290>.

Soumyasch (2007). The .NET Framework stack [Online image]. Wikipedia. <https://en.wikipedia.org/wiki/.NET_Framework_version_history#/media/File:DotNet.svg>.

Xu, J. (2010). Practical WPF Charts and Graphics. In *Practical WPF Charts and Graphics*. Apress. <https://doi.org/10.1007/978-1-4302-2482-2>.

Youssef, A. (1999). Image downsampling and upsampling methods. *National Institute of Standards and Technology*.

# Appendix 1 – Project Proposal

## Computing Degree Project Proposal

|  |  |
| --- | --- |
| **Student Name: Lili Veszeli** |  |
| **Course: Computer Games Development** |  |
| **Project Title: Cross Stitch Pattern Generator** |  |

## Project Context

I would like to create a computer software that could convert a picture to a cross stitch pattern. There are other software like this available online for free or for a price. (Chestnut Pens, 2020; Klein, 2017) This would be an interesting project that involves all the techniques I learnt at university.

## Specific Objectives

* Generate a pattern based on a picture
* Choose desired number of colours
* Choose size of pattern
* Edit pattern (for example change colour or draw on it)
* Save the pattern at the end

## Potential Ethical or Legal Issues

Copyright issues. The software might be too similar to one previously created. However it would not be illegal to do it for an educational purpose (Copyright, Designs and Patents Act 1988).

Some users could abuse the software to make patterns from copyrighted images/characters and sell them, which would be illegal.

Some users could make patterns with disturbing content, which might be seen by children if the patterns are sold.

## Resources

Computer – available at university and also own laptop

Visual Studio – available.

UI library for C++ - there are some free options, like JUCE (Robinson, 2013)

OpenCV – available

DirectX – if needed available

## Potential Commercial Considerations - Estimated costs and benefits

The software would be mainly used by people who enjoy cross stitching, which can be done any time. Hobbies including arts and crafts are becoming more popular (Kelly et al, 2020), a huge rise was experienced during lockdown (The Guardian, 2020). There are people who enjoy it from all generations, so I am confident that it will continue to be practiced and hence the program will be used. With the help of the application, it is possible to make patterns from pop culture, so it might help to make it more popular among young people.

Cost of programming the software: it will take around 30 weeks (academic year), so the salary for a programmer for that period would be approximately £12750.

The software would be sold for individuals for a low price, £10-15, because most similar applications start at £50, and I think a casual cross stitcher would be more likely to buy a product with a lower price.

All similar free applications have a bad quality UI, so by making the UI of my program clean and compelling it will be more attractive for users.

It would need to be advertised in some way, so that would add more to the costs depending which way it is advertised. If used Facebook advertisement, it would cost £300 for a month.

## Proposed Approach

Writing Technical Plan – 1 week

Researching how to load an image using C++ and code it – 1 week

Researching how to convert the pixels in the image to the squares in the pattern and code it – 5 weeks

Researching how to draw the grid over the squares and code it – 1 week

Researching how to edit the pattern and code it – 10 weeks

Researching how to make UI and code it – 3 weeks

Researching how to save file and code it – 1 week

Testing the application – 1 week (Testing would be involved every stage, but a more comprehensive testing period would be at the end)

Writing report - 4 weeks

## References

Chestnut Pens (2020) Cross Stitch Chart Creation Program. Retrieved from: <http://www.chestnutpens.co.uk/misc/index.html>

Copyright, Designs and Patents Act 1988, c.3. Retrieved from: https://www.legislation.gov.uk/ukpga/1988/48/

Kaehler, A., & Bradski, G. (2016). Learning OpenCV 3: computer vision in C++ with the OpenCV library. " O'Reilly Media, Inc.".

Kelly, C. M., Strauss, K., Arnold, J., & Stride, C. (2020). The relationship between leisure activities and psychological resources that support a sustainable career: The role of leisure seriousness and work-leisure similarity. Journal of Vocational Behavior, 117, 103340. Retrieved from: https://doi.org/https://doi.org/10.1016/j.jvb.2019.103340

Klein, T. (2017) Cstitch. Retrieved from: <http://cstitch.sourceforge.net/index.shtml>

OpenCV (2019) Load, Modify, and Save an Image. Retrieved from: <https://docs.opencv.org/2.4/doc/tutorials/introduction/load_save_image/load_save_image.html>

Robinson, M. (2013). Getting started with JUCE. Packt Publishing Ltd.

The Guardian (2020) Hobbycraft reports 200% boom in online sales since start of pandemic. Retrieved from: https://www.theguardian.com/business/2020/aug/03/hobbycraft-reports-boom-in-online-sales-since-start-of-pandemic-crafts-coronavirus.

Wang, M., & Lai, C. H. (2016). A concise introduction to image processing using C++. CRC Press.

Appendix 2 – Technical Plan

## Computing Project Technical Plan

**Name:** Lili Veszeli

**Course:** Computer Games Development

**Supervisory Team:** Games Team

## Title

Cross Stitch Pattern Generator.

Summary

I will create a computer application that will generate a cross stitch pattern based on an image. The user would be able to modify the pattern, for example change colours, change stitch location, change overall size. The pattern can be saved and printed. I will also have to a make a User Interface.

The difficult part in this project will be figuring out how to edit the pattern, making sure that the pattern looks right and uses the correct colours. I will need to research different methods to create these features. Working with C# will also be challenging since I am more familiar with C++.

I will use the Waterfall model as my development methodology, since we have set deadlines and once we submitted the required material we can’t change it. Some tools from Agile will be used too, like Moscow; this can make sure that all the essential features will be complete (Waters, 2009). Testing will be done too to make sure the program works as expected.

## Deliverables

The main product of this project will be a Windows based computer software that will be able to generate a cross stitch pattern from a picture. Other deliverables will be the report and poster.

## Constraints

The most serious constraint is the deadlines. It might be the case that the application won’t have all the features because of the limited time available for development.

Another constraint could be lack of resources. I might not find specific enough resources about how to create different features for my project.

The methodology used in my project could be a constraint. Using the Agile methodology is not possible since we can’t change our project after submitting the proposal. This might not be flexible enough for the project.

## Key Problems

The main difficulties will be

* to figure out how to edit the generated cross stitch pattern. I am not familiar enough with WPF to know how it could be done, so I will need to research it more.
* to design an efficient and clean user interface. I will need to research examples for it and experiment with my UI.
* to figure out how to work with pixels in WPF.

## System and Work Outline

The project will be developed will be developed using WPF in C# and XAML. The UI component will be made using WPF’s built in drag and drop UI editor. This can be further customized with XAML. The behaviour of the UI and the backend will be programmed in C#. Visual Studio will be used for the development environment. GitHub will be used for version control and backup.

Features of the program using Moscow prioritization:

|  |
| --- |
| **Must have:** |
| * A Windows application that can convert a picture’s pixels to bigger squares. * Basic UI. * A grid over the picture/pattern. * The ability to save it and print it. |
| **Should have:** |
| * The ability to resize the pattern. * More advanced UI. * Saving it in a PDF. * The ability to edit the pattern. * The ability to reduce the number of colours used in the pattern. |
| **Could have:** |
| * The ability to use real thread colours. * Zoom in while editing. |
| **Won’t have (but Would like in the future):** |
| * Other types of stitches. * Representation of how the pattern will look once finished. * Blended colours. |

(Agile Business Consortium, 2019)

Regarding WPF, I will need to research how images could be loaded in, how to make a modern UI and how to manipulate graphics (Sells & Griffiths, 2007).

I will also need to research how to navigate Windows folders, save and print files (Microsoft Docs., 2020).

## Project Activities



## Risk Analysis

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Severity** | **Likelihood** | **Action** |
| My computer breaks | Severe | Small | Get it fixed – would hinder development by a few days.  Alternatively, work in the games lab – good solution, but might be difficult because of the limited availability this year. |
| Not being able to implement every planned feature. | Low | High | To be able to demonstrate my program it only needs the most important features, which are highly likely to be implemented. Using Moscow will help to ensure this. |
| Loss of data | Severe | Low | Making copies of the project every time I add to it, and storing it on a cloud based server. |

## Options

As an alternative to the Waterfall method, Agile could be used. However, because the deadlines for the deliverables are already set it makes the development less flexible. Main features cannot be changed either. Since I know what kind of features I want, I won’t have to do major changes to the program. It is also harder to use agile techniques if developing alone. Because of these constraints the best suited project management methodology for this project is Waterfall (Balaji & Murugaiyan ,2012).

An alternative I considered to develop the program in was Windows forms. However, WPF is a more modern and more flexible version of Windows forms, with more customization options (“WPF vs. WinForms”, 2020*)*. In WPF the UI editor is in XAML, with a drag and drop menu.

Another option could be DirectX 11. It would need a Windows library to be able to make the UI and it would be unnecessarily complicated to make the pattern using shaders.

Windows is the target environment. This program can be used most effectively on a computer, in a mobile app version the user wouldn’t be able to use the features properly. Later on, the project could be expanded to macOS and Linux, but this is not part of the objectives now because of the time limitation.

## Potential Ethical or Legal Issues

Legal issues might arise regarding the copyright law (Copyright, Designs and Patents Act 1988). If the program turns out to be too similar to one previously made, it could be a problem. However, this can be avoided by writing original code, not copying code that isn’t allowed to be freely used, and designing my own UI and mechanics of the application.

## Commercial Analysis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Factor name** | **Description** | **Is this a cost or a benefit** | **Estimated Amount** | **Estimate of when paid** |
| Hardware | Computer to develop the project on. | Cost | Cost of my laptop - £900 | Start of the project |
| Software | Windows operating system | Cost | Cost of Windows 10 Pro - £220 | Start of the project |
| Salary | Salary for developing the program | Benefit | Salary for 30 weeks (academic year) based on average junior games developer salary - £12750  (Prospects, 2020) | Every month |
| Advertisement | Cost to advertise the product to the targeted audience | Cost | Facebook advertisement for 2 months –  £600 | End of project |
| Price | Price of the product | Benefit | £15 every time it’s purchased | End of project |

## Employability Contribution

This project will help me get a better understanding of making software for the public. It will improve my skill at UI design and my coding abilities in C# and XAML. I will gain experience in programming a Windows application. After writing the report I will be better at written communication.

The project can be expanded with a lot more features and could be marketed.

## References

Agile Business Consortium. (2019). *Chapter 10: Moscow Prioritisation*. Retrieved from: <https://www.agilebusiness.org/page/ProjectFramework_10_MoSCoWPrioritisation>

Balaji, S., & Murugaiyan, M. S. (2012). Waterfall vs. V-Model vs. Agile: A comparative study on SDLC. *International Journal of Information Technology and Business Management*, *2*(1), 26-30.

Copyright, Designs and Patents Act 1988, c.3. Retrieved from: <https://www.legislation.gov.uk/ukpga/1988/48/>

*Game developer job profile | Prospects.ac.uk*. (2020). Retrieved October 26, 2020, from <https://www.prospects.ac.uk/job-profiles/game-developer>

*PrintDocument.Print Method (System.Drawing.Printing) | Microsoft Docs*. (2020). Retrieved October 26, 2020, from https://docs.microsoft.com/en-us/dotnet/api/system.drawing.printing.printdocument.print?view=dotnet-plat-ext-3.1

Sells, C., & Griffiths, I. (2007). *Programming WPF: Building Windows UI with Windows Presentation Foundation*. " O'Reilly Media, Inc.".

Waters, K. (2009). Prioritization using moscow. *Agile Planning*, *12*, 31.

*WPF vs. WinForms - The complete WPF tutorial*. (2020). Retrieved October 26, 2020, from https://www.wpf-tutorial.com/about-wpf/wpf-vs-winforms/

# Appendix 3 – Software Licenses

## MIT License

Copyright <YEAR> <COPYRIGHT HOLDER>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## Apache License

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction,

and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by

the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all

other entities that control, are controlled by, or are under common

control with that entity. For the purposes of this definition,

"control" means (i) the power, direct or indirect, to cause the

direction or management of such entity, whether by contract or

otherwise, or (ii) ownership of fifty percent (50%) or more of the

outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity

exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications,

including but not limited to software source code, documentation

source, and configuration files.

"Object" form shall mean any form resulting from mechanical

transformation or translation of a Source form, including but

not limited to compiled object code, generated documentation,

and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or

Object form, made available under the License, as indicated by a

copyright notice that is included in or attached to the work

(an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object

form, that is based on (or derived from) the Work and for which the

editorial revisions, annotations, elaborations, or other modifications

represent, as a whole, an original work of authorship. For the purposes

of this License, Derivative Works shall not include works that remain

separable from, or merely link (or bind by name) to the interfaces of,

the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including

the original version of the Work and any modifications or additions

to that Work or Derivative Works thereof, that is intentionally

submitted to Licensor for inclusion in the Work by the copyright owner

or by an individual or Legal Entity authorized to submit on behalf of

the copyright owner. For the purposes of this definition, "submitted"

means any form of electronic, verbal, or written communication sent

to the Licensor or its representatives, including but not limited to

communication on electronic mailing lists, source code control systems,

and issue tracking systems that are managed by, or on behalf of, the

Licensor for the purpose of discussing and improving the Work, but

excluding communication that is conspicuously marked or otherwise

designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity

on behalf of whom a Contribution has been received by Licensor and

subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of

this License, each Contributor hereby grants to You a perpetual,

worldwide, non-exclusive, no-charge, royalty-free, irrevocable

copyright license to reproduce, prepare Derivative Works of,

publicly display, publicly perform, sublicense, and distribute the

Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of

this License, each Contributor hereby grants to You a perpetual,

worldwide, non-exclusive, no-charge, royalty-free, irrevocable

(except as stated in this section) patent license to make, have made,

use, offer to sell, sell, import, and otherwise transfer the Work,

where such license applies only to those patent claims licensable

by such Contributor that are necessarily infringed by their

Contribution(s) alone or by combination of their Contribution(s)

with the Work to which such Contribution(s) was submitted. If You

institute patent litigation against any entity (including a

cross-claim or counterclaim in a lawsuit) alleging that the Work

or a Contribution incorporated within the Work constitutes direct

or contributory patent infringement, then any patent licenses

granted to You under this License for that Work shall terminate

as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the

Work or Derivative Works thereof in any medium, with or without

modifications, and in Source or Object form, provided that You

meet the following conditions:

(a) You must give any other recipients of the Work or

Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices

stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works

that You distribute, all copyright, patent, trademark, and

attribution notices from the Source form of the Work,

excluding those notices that do not pertain to any part of

the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its

distribution, then any Derivative Works that You distribute must

include a readable copy of the attribution notices contained

within such NOTICE file, excluding those notices that do not

pertain to any part of the Derivative Works, in at least one

of the following places: within a NOTICE text file distributed

as part of the Derivative Works; within the Source form or

documentation, if provided along with the Derivative Works; or,

within a display generated by the Derivative Works, if and

wherever such third-party notices normally appear. The contents

of the NOTICE file are for informational purposes only and

do not modify the License. You may add Your own attribution

notices within Derivative Works that You distribute, alongside

or as an addendum to the NOTICE text from the Work, provided

that such additional attribution notices cannot be construed

as modifying the License.

You may add Your own copyright statement to Your modifications and

may provide additional or different license terms and conditions

for use, reproduction, or distribution of Your modifications, or

for any such Derivative Works as a whole, provided Your use,

reproduction, and distribution of the Work otherwise complies with

the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise,

any Contribution intentionally submitted for inclusion in the Work

by You to the Licensor shall be under the terms and conditions of

this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify

the terms of any separate license agreement you may have executed

with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade

names, trademarks, service marks, or product names of the Licensor,

except as required for reasonable and customary use in describing the

origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or

agreed to in writing, Licensor provides the Work (and each

Contributor provides its Contributions) on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or

implied, including, without limitation, any warranties or conditions

of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A

PARTICULAR PURPOSE. You are solely responsible for determining the

appropriateness of using or redistributing the Work and assume any

risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,

whether in tort (including negligence), contract, or otherwise,

unless required by applicable law (such as deliberate and grossly

negligent acts) or agreed to in writing, shall any Contributor be

liable to You for damages, including any direct, indirect, special,

incidental, or consequential damages of any character arising as a

result of this License or out of the use or inability to use the

Work (including but not limited to damages for loss of goodwill,

work stoppage, computer failure or malfunction, or any and all

other commercial damages or losses), even if such Contributor

has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing

the Work or Derivative Works thereof, You may choose to offer,

and charge a fee for, acceptance of support, warranty, indemnity,

or other liability obligations and/or rights consistent with this

License. However, in accepting such obligations, You may act only

on Your own behalf and on Your sole responsibility, not on behalf

of any other Contributor, and only if You agree to indemnify,

defend, and hold each Contributor harmless for any liability

incurred by, or claims asserted against, such Contributor by reason

of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following

boilerplate notice, with the fields enclosed by brackets "[]"

replaced with your own identifying information. (Don't include

the brackets!) The text should be enclosed in the appropriate

comment syntax for the file format. We also recommend that a

file or class name and description of purpose be included on the

same "printed page" as the copyright notice for easier

identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

## GNU LGPL v3 License

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies

of this license document, but changing it is not allowed.

|  |
| --- |
| Preamble |
|  |  |
|  | The GNU General Public License is a free, copyleft license for |
|  | software and other kinds of works. |
|  |  |
|  | The licenses for most software and other practical works are designed |
|  | to take away your freedom to share and change the works. By contrast, |
|  | the GNU General Public License is intended to guarantee your freedom to |
|  | share and change all versions of a program--to make sure it remains free |
|  | software for all its users. We, the Free Software Foundation, use the |
|  | GNU General Public License for most of our software; it applies also to |
|  | any other work released this way by its authors. You can apply it to |
|  | your programs, too. |
|  |  |
|  | When we speak of free software, we are referring to freedom, not |
|  | price. Our General Public Licenses are designed to make sure that you |
|  | have the freedom to distribute copies of free software (and charge for |
|  | them if you wish), that you receive source code or can get it if you |
|  | want it, that you can change the software or use pieces of it in new |
|  | free programs, and that you know you can do these things. |
|  |  |
|  | To protect your rights, we need to prevent others from denying you |
|  | these rights or asking you to surrender the rights. Therefore, you have |
|  | certain responsibilities if you distribute copies of the software, or if |
|  | you modify it: responsibilities to respect the freedom of others. |
|  |  |
|  | For example, if you distribute copies of such a program, whether |
|  | gratis or for a fee, you must pass on to the recipients the same |
|  | freedoms that you received. You must make sure that they, too, receive |
|  | or can get the source code. And you must show them these terms so they |
|  | know their rights. |
|  |  |
|  | Developers that use the GNU GPL protect your rights with two steps: |
|  | (1) assert copyright on the software, and (2) offer you this License |
|  | giving you legal permission to copy, distribute and/or modify it. |
|  |  |
|  | For the developers' and authors' protection, the GPL clearly explains |
|  | that there is no warranty for this free software. For both users' and |
|  | authors' sake, the GPL requires that modified versions be marked as |
|  | changed, so that their problems will not be attributed erroneously to |
|  | authors of previous versions. |
|  |  |
|  | Some devices are designed to deny users access to install or run |
|  | modified versions of the software inside them, although the manufacturer |
|  | can do so. This is fundamentally incompatible with the aim of |
|  | protecting users' freedom to change the software. The systematic |
|  | pattern of such abuse occurs in the area of products for individuals to |
|  | use, which is precisely where it is most unacceptable. Therefore, we |
|  | have designed this version of the GPL to prohibit the practice for those |
|  | products. If such problems arise substantially in other domains, we |
|  | stand ready to extend this provision to those domains in future versions |
|  | of the GPL, as needed to protect the freedom of users. |
|  |  |
|  | Finally, every program is threatened constantly by software patents. |
|  | States should not allow patents to restrict development and use of |
|  | software on general-purpose computers, but in those that do, we wish to |
|  | avoid the special danger that patents applied to a free program could |
|  | make it effectively proprietary. To prevent this, the GPL assures that |
|  | patents cannot be used to render the program non-free. |
|  |  |
|  | The precise terms and conditions for copying, distribution and |
|  | modification follow. |
|  |  |
|  | TERMS AND CONDITIONS |
|  |  |
|  | 0. Definitions. |
|  |  |
|  | "This License" refers to version 3 of the GNU General Public License. |
|  |  |
|  | "Copyright" also means copyright-like laws that apply to other kinds of |
|  | works, such as semiconductor masks. |
|  |  |
|  | "The Program" refers to any copyrightable work licensed under this |
|  | License. Each licensee is addressed as "you". "Licensees" and |
|  | "recipients" may be individuals or organizations. |
|  |  |
|  | To "modify" a work means to copy from or adapt all or part of the work |
|  | in a fashion requiring copyright permission, other than the making of an |
|  | exact copy. The resulting work is called a "modified version" of the |
|  | earlier work or a work "based on" the earlier work. |
|  |  |
|  | A "covered work" means either the unmodified Program or a work based |
|  | on the Program. |
|  |  |
|  | To "propagate" a work means to do anything with it that, without |
|  | permission, would make you directly or secondarily liable for |
|  | infringement under applicable copyright law, except executing it on a |
|  | computer or modifying a private copy. Propagation includes copying, |
|  | distribution (with or without modification), making available to the |
|  | public, and in some countries other activities as well. |
|  |  |
|  | To "convey" a work means any kind of propagation that enables other |
|  | parties to make or receive copies. Mere interaction with a user through |
|  | a computer network, with no transfer of a copy, is not conveying. |
|  |  |
|  | An interactive user interface displays "Appropriate Legal Notices" |
|  | to the extent that it includes a convenient and prominently visible |
|  | feature that (1) displays an appropriate copyright notice, and (2) |
|  | tells the user that there is no warranty for the work (except to the |
|  | extent that warranties are provided), that licensees may convey the |
|  | work under this License, and how to view a copy of this License. If |
|  | the interface presents a list of user commands or options, such as a |
|  | menu, a prominent item in the list meets this criterion. |
|  |  |
|  | 1. Source Code. |
|  |  |
|  | The "source code" for a work means the preferred form of the work |
|  | for making modifications to it. "Object code" means any non-source |
|  | form of a work. |
|  |  |
|  | A "Standard Interface" means an interface that either is an official |
|  | standard defined by a recognized standards body, or, in the case of |
|  | interfaces specified for a particular programming language, one that |
|  | is widely used among developers working in that language. |
|  |  |
|  | The "System Libraries" of an executable work include anything, other |
|  | than the work as a whole, that (a) is included in the normal form of |
|  | packaging a Major Component, but which is not part of that Major |
|  | Component, and (b) serves only to enable use of the work with that |
|  | Major Component, or to implement a Standard Interface for which an |
|  | implementation is available to the public in source code form. A |
|  | "Major Component", in this context, means a major essential component |
|  | (kernel, window system, and so on) of the specific operating system |
|  | (if any) on which the executable work runs, or a compiler used to |
|  | produce the work, or an object code interpreter used to run it. |
|  |  |
|  | The "Corresponding Source" for a work in object code form means all |
|  | the source code needed to generate, install, and (for an executable |
|  | work) run the object code and to modify the work, including scripts to |
|  | control those activities. However, it does not include the work's |
|  | System Libraries, or general-purpose tools or generally available free |
|  | programs which are used unmodified in performing those activities but |
|  | which are not part of the work. For example, Corresponding Source |
|  | includes interface definition files associated with source files for |
|  | the work, and the source code for shared libraries and dynamically |
|  | linked subprograms that the work is specifically designed to require, |
|  | such as by intimate data communication or control flow between those |
|  | subprograms and other parts of the work. |
|  |  |
|  | The Corresponding Source need not include anything that users |
|  | can regenerate automatically from other parts of the Corresponding |
|  | Source. |
|  |  |
|  | The Corresponding Source for a work in source code form is that |
|  | same work. |
|  |  |
|  | 2. Basic Permissions. |
|  |  |
|  | All rights granted under this License are granted for the term of |
|  | copyright on the Program, and are irrevocable provided the stated |
|  | conditions are met. This License explicitly affirms your unlimited |
|  | permission to run the unmodified Program. The output from running a |
|  | covered work is covered by this License only if the output, given its |
|  | content, constitutes a covered work. This License acknowledges your |
|  | rights of fair use or other equivalent, as provided by copyright law. |
|  |  |
|  | You may make, run and propagate covered works that you do not |
|  | convey, without conditions so long as your license otherwise remains |
|  | in force. You may convey covered works to others for the sole purpose |
|  | of having them make modifications exclusively for you, or provide you |
|  | with facilities for running those works, provided that you comply with |
|  | the terms of this License in conveying all material for which you do |
|  | not control copyright. Those thus making or running the covered works |
|  | for you must do so exclusively on your behalf, under your direction |
|  | and control, on terms that prohibit them from making any copies of |
|  | your copyrighted material outside their relationship with you. |
|  |  |
|  | Conveying under any other circumstances is permitted solely under |
|  | the conditions stated below. Sublicensing is not allowed; section 10 |
|  | makes it unnecessary. |
|  |  |
|  | 3. Protecting Users' Legal Rights From Anti-Circumvention Law. |
|  |  |
|  | No covered work shall be deemed part of an effective technological |
|  | measure under any applicable law fulfilling obligations under article |
|  | 11 of the WIPO copyright treaty adopted on 20 December 1996, or |
|  | similar laws prohibiting or restricting circumvention of such |
|  | measures. |
|  |  |
|  | When you convey a covered work, you waive any legal power to forbid |
|  | circumvention of technological measures to the extent such circumvention |
|  | is effected by exercising rights under this License with respect to |
|  | the covered work, and you disclaim any intention to limit operation or |
|  | modification of the work as a means of enforcing, against the work's |
|  | users, your or third parties' legal rights to forbid circumvention of |
|  | technological measures. |
|  |  |
|  | 4. Conveying Verbatim Copies. |
|  |  |
|  | You may convey verbatim copies of the Program's source code as you |
|  | receive it, in any medium, provided that you conspicuously and |
|  | appropriately publish on each copy an appropriate copyright notice; |
|  | keep intact all notices stating that this License and any |
|  | non-permissive terms added in accord with section 7 apply to the code; |
|  | keep intact all notices of the absence of any warranty; and give all |
|  | recipients a copy of this License along with the Program. |
|  |  |
|  | You may charge any price or no price for each copy that you convey, |
|  | and you may offer support or warranty protection for a fee. |
|  |  |
|  | 5. Conveying Modified Source Versions. |
|  |  |
|  | You may convey a work based on the Program, or the modifications to |
|  | produce it from the Program, in the form of source code under the |
|  | terms of section 4, provided that you also meet all of these conditions: |
|  |  |
|  | a) The work must carry prominent notices stating that you modified |
|  | it, and giving a relevant date. |
|  |  |
|  | b) The work must carry prominent notices stating that it is |
|  | released under this License and any conditions added under section |
|  | 7. This requirement modifies the requirement in section 4 to |
|  | "keep intact all notices". |
|  |  |
|  | c) You must license the entire work, as a whole, under this |
|  | License to anyone who comes into possession of a copy. This |
|  | License will therefore apply, along with any applicable section 7 |
|  | additional terms, to the whole of the work, and all its parts, |
|  | regardless of how they are packaged. This License gives no |
|  | permission to license the work in any other way, but it does not |
|  | invalidate such permission if you have separately received it. |
|  |  |
|  | d) If the work has interactive user interfaces, each must display |
|  | Appropriate Legal Notices; however, if the Program has interactive |
|  | interfaces that do not display Appropriate Legal Notices, your |
|  | work need not make them do so. |
|  |  |
|  | A compilation of a covered work with other separate and independent |
|  | works, which are not by their nature extensions of the covered work, |
|  | and which are not combined with it such as to form a larger program, |
|  | in or on a volume of a storage or distribution medium, is called an |
|  | "aggregate" if the compilation and its resulting copyright are not |
|  | used to limit the access or legal rights of the compilation's users |
|  | beyond what the individual works permit. Inclusion of a covered work |
|  | in an aggregate does not cause this License to apply to the other |
|  | parts of the aggregate. |
|  |  |
|  | 6. Conveying Non-Source Forms. |
|  |  |
|  | You may convey a covered work in object code form under the terms |
|  | of sections 4 and 5, provided that you also convey the |
|  | machine-readable Corresponding Source under the terms of this License, |
|  | in one of these ways: |
|  |  |
|  | a) Convey the object code in, or embodied in, a physical product |
|  | (including a physical distribution medium), accompanied by the |
|  | Corresponding Source fixed on a durable physical medium |
|  | customarily used for software interchange. |
|  |  |
|  | b) Convey the object code in, or embodied in, a physical product |
|  | (including a physical distribution medium), accompanied by a |
|  | written offer, valid for at least three years and valid for as |
|  | long as you offer spare parts or customer support for that product |
|  | model, to give anyone who possesses the object code either (1) a |
|  | copy of the Corresponding Source for all the software in the |
|  | product that is covered by this License, on a durable physical |
|  | medium customarily used for software interchange, for a price no |
|  | more than your reasonable cost of physically performing this |
|  | conveying of source, or (2) access to copy the |
|  | Corresponding Source from a network server at no charge. |
|  |  |
|  | c) Convey individual copies of the object code with a copy of the |
|  | written offer to provide the Corresponding Source. This |
|  | alternative is allowed only occasionally and noncommercially, and |
|  | only if you received the object code with such an offer, in accord |
|  | with subsection 6b. |
|  |  |
|  | d) Convey the object code by offering access from a designated |
|  | place (gratis or for a charge), and offer equivalent access to the |
|  | Corresponding Source in the same way through the same place at no |
|  | further charge. You need not require recipients to copy the |
|  | Corresponding Source along with the object code. If the place to |
|  | copy the object code is a network server, the Corresponding Source |
|  | may be on a different server (operated by you or a third party) |
|  | that supports equivalent copying facilities, provided you maintain |
|  | clear directions next to the object code saying where to find the |
|  | Corresponding Source. Regardless of what server hosts the |
|  | Corresponding Source, you remain obligated to ensure that it is |
|  | available for as long as needed to satisfy these requirements. |
|  |  |
|  | e) Convey the object code using peer-to-peer transmission, provided |
|  | you inform other peers where the object code and Corresponding |
|  | Source of the work are being offered to the general public at no |
|  | charge under subsection 6d. |
|  |  |
|  | A separable portion of the object code, whose source code is excluded |
|  | from the Corresponding Source as a System Library, need not be |
|  | included in conveying the object code work. |
|  |  |
|  | A "User Product" is either (1) a "consumer product", which means any |
|  | tangible personal property which is normally used for personal, family, |
|  | or household purposes, or (2) anything designed or sold for incorporation |
|  | into a dwelling. In determining whether a product is a consumer product, |
|  | doubtful cases shall be resolved in favor of coverage. For a particular |
|  | product received by a particular user, "normally used" refers to a |
|  | typical or common use of that class of product, regardless of the status |
|  | of the particular user or of the way in which the particular user |
|  | actually uses, or expects or is expected to use, the product. A product |
|  | is a consumer product regardless of whether the product has substantial |
|  | commercial, industrial or non-consumer uses, unless such uses represent |
|  | the only significant mode of use of the product. |
|  |  |
|  | "Installation Information" for a User Product means any methods, |
|  | procedures, authorization keys, or other information required to install |
|  | and execute modified versions of a covered work in that User Product from |
|  | a modified version of its Corresponding Source. The information must |
|  | suffice to ensure that the continued functioning of the modified object |
|  | code is in no case prevented or interfered with solely because |
|  | modification has been made. |
|  |  |
|  | If you convey an object code work under this section in, or with, or |
|  | specifically for use in, a User Product, and the conveying occurs as |
|  | part of a transaction in which the right of possession and use of the |
|  | User Product is transferred to the recipient in perpetuity or for a |
|  | fixed term (regardless of how the transaction is characterized), the |
|  | Corresponding Source conveyed under this section must be accompanied |
|  | by the Installation Information. But this requirement does not apply |
|  | if neither you nor any third party retains the ability to install |
|  | modified object code on the User Product (for example, the work has |
|  | been installed in ROM). |
|  |  |
|  | The requirement to provide Installation Information does not include a |
|  | requirement to continue to provide support service, warranty, or updates |
|  | for a work that has been modified or installed by the recipient, or for |
|  | the User Product in which it has been modified or installed. Access to a |
|  | network may be denied when the modification itself materially and |
|  | adversely affects the operation of the network or violates the rules and |
|  | protocols for communication across the network. |
|  |  |
|  | Corresponding Source conveyed, and Installation Information provided, |
|  | in accord with this section must be in a format that is publicly |
|  | documented (and with an implementation available to the public in |
|  | source code form), and must require no special password or key for |
|  | unpacking, reading or copying. |
|  |  |
|  | 7. Additional Terms. |
|  |  |
|  | "Additional permissions" are terms that supplement the terms of this |
|  | License by making exceptions from one or more of its conditions. |
|  | Additional permissions that are applicable to the entire Program shall |
|  | be treated as though they were included in this License, to the extent |
|  | that they are valid under applicable law. If additional permissions |
|  | apply only to part of the Program, that part may be used separately |
|  | under those permissions, but the entire Program remains governed by |
|  | this License without regard to the additional permissions. |
|  |  |
|  | When you convey a copy of a covered work, you may at your option |
|  | remove any additional permissions from that copy, or from any part of |
|  | it. (Additional permissions may be written to require their own |
|  | removal in certain cases when you modify the work.) You may place |
|  | additional permissions on material, added by you to a covered work, |
|  | for which you have or can give appropriate copyright permission. |
|  |  |
|  | Notwithstanding any other provision of this License, for material you |
|  | add to a covered work, you may (if authorized by the copyright holders of |
|  | that material) supplement the terms of this License with terms: |
|  |  |
|  | a) Disclaiming warranty or limiting liability differently from the |
|  | terms of sections 15 and 16 of this License; or |
|  |  |
|  | b) Requiring preservation of specified reasonable legal notices or |
|  | author attributions in that material or in the Appropriate Legal |
|  | Notices displayed by works containing it; or |
|  |  |
|  | c) Prohibiting misrepresentation of the origin of that material, or |
|  | requiring that modified versions of such material be marked in |
|  | reasonable ways as different from the original version; or |
|  |  |
|  | d) Limiting the use for publicity purposes of names of licensors or |
|  | authors of the material; or |
|  |  |
|  | e) Declining to grant rights under trademark law for use of some |
|  | trade names, trademarks, or service marks; or |
|  |  |
|  | f) Requiring indemnification of licensors and authors of that |
|  | material by anyone who conveys the material (or modified versions of |
|  | it) with contractual assumptions of liability to the recipient, for |
|  | any liability that these contractual assumptions directly impose on |
|  | those licensors and authors. |
|  |  |
|  | All other non-permissive additional terms are considered "further |
|  | restrictions" within the meaning of section 10. If the Program as you |
|  | received it, or any part of it, contains a notice stating that it is |
|  | governed by this License along with a term that is a further |
|  | restriction, you may remove that term. If a license document contains |
|  | a further restriction but permits relicensing or conveying under this |
|  | License, you may add to a covered work material governed by the terms |
|  | of that license document, provided that the further restriction does |
|  | not survive such relicensing or conveying. |
|  |  |
|  | If you add terms to a covered work in accord with this section, you |
|  | must place, in the relevant source files, a statement of the |
|  | additional terms that apply to those files, or a notice indicating |
|  | where to find the applicable terms. |
|  |  |
|  | Additional terms, permissive or non-permissive, may be stated in the |
|  | form of a separately written license, or stated as exceptions; |
|  | the above requirements apply either way. |
|  |  |
|  | 8. Termination. |
|  |  |
|  | You may not propagate or modify a covered work except as expressly |
|  | provided under this License. Any attempt otherwise to propagate or |
|  | modify it is void, and will automatically terminate your rights under |
|  | this License (including any patent licenses granted under the third |
|  | paragraph of section 11). |
|  |  |
|  | However, if you cease all violation of this License, then your |
|  | license from a particular copyright holder is reinstated (a) |
|  | provisionally, unless and until the copyright holder explicitly and |
|  | finally terminates your license, and (b) permanently, if the copyright |
|  | holder fails to notify you of the violation by some reasonable means |
|  | prior to 60 days after the cessation. |
|  |  |
|  | Moreover, your license from a particular copyright holder is |
|  | reinstated permanently if the copyright holder notifies you of the |
|  | violation by some reasonable means, this is the first time you have |
|  | received notice of violation of this License (for any work) from that |
|  | copyright holder, and you cure the violation prior to 30 days after |
|  | your receipt of the notice. |
|  |  |
|  | Termination of your rights under this section does not terminate the |
|  | licenses of parties who have received copies or rights from you under |
|  | this License. If your rights have been terminated and not permanently |
|  | reinstated, you do not qualify to receive new licenses for the same |
|  | material under section 10. |
|  |  |
|  | 9. Acceptance Not Required for Having Copies. |
|  |  |
|  | You are not required to accept this License in order to receive or |
|  | run a copy of the Program. Ancillary propagation of a covered work |
|  | occurring solely as a consequence of using peer-to-peer transmission |
|  | to receive a copy likewise does not require acceptance. However, |
|  | nothing other than this License grants you permission to propagate or |
|  | modify any covered work. These actions infringe copyright if you do |
|  | not accept this License. Therefore, by modifying or propagating a |
|  | covered work, you indicate your acceptance of this License to do so. |
|  |  |
|  | 10. Automatic Licensing of Downstream Recipients. |
|  |  |
|  | Each time you convey a covered work, the recipient automatically |
|  | receives a license from the original licensors, to run, modify and |
|  | propagate that work, subject to this License. You are not responsible |
|  | for enforcing compliance by third parties with this License. |
|  |  |
|  | An "entity transaction" is a transaction transferring control of an |
|  | organization, or substantially all assets of one, or subdividing an |
|  | organization, or merging organizations. If propagation of a covered |
|  | work results from an entity transaction, each party to that |
|  | transaction who receives a copy of the work also receives whatever |
|  | licenses to the work the party's predecessor in interest had or could |
|  | give under the previous paragraph, plus a right to possession of the |
|  | Corresponding Source of the work from the predecessor in interest, if |
|  | the predecessor has it or can get it with reasonable efforts. |
|  |  |
|  | You may not impose any further restrictions on the exercise of the |
|  | rights granted or affirmed under this License. For example, you may |
|  | not impose a license fee, royalty, or other charge for exercise of |
|  | rights granted under this License, and you may not initiate litigation |
|  | (including a cross-claim or counterclaim in a lawsuit) alleging that |
|  | any patent claim is infringed by making, using, selling, offering for |
|  | sale, or importing the Program or any portion of it. |
|  |  |
|  | 11. Patents. |
|  |  |
|  | A "contributor" is a copyright holder who authorizes use under this |
|  | License of the Program or a work on which the Program is based. The |
|  | work thus licensed is called the contributor's "contributor version". |
|  |  |
|  | A contributor's "essential patent claims" are all patent claims |
|  | owned or controlled by the contributor, whether already acquired or |
|  | hereafter acquired, that would be infringed by some manner, permitted |
|  | by this License, of making, using, or selling its contributor version, |
|  | but do not include claims that would be infringed only as a |
|  | consequence of further modification of the contributor version. For |
|  | purposes of this definition, "control" includes the right to grant |
|  | patent sublicenses in a manner consistent with the requirements of |
|  | this License. |
|  |  |
|  | Each contributor grants you a non-exclusive, worldwide, royalty-free |
|  | patent license under the contributor's essential patent claims, to |
|  | make, use, sell, offer for sale, import and otherwise run, modify and |
|  | propagate the contents of its contributor version. |
|  |  |
|  | In the following three paragraphs, a "patent license" is any express |
|  | agreement or commitment, however denominated, not to enforce a patent |
|  | (such as an express permission to practice a patent or covenant not to |
|  | sue for patent infringement). To "grant" such a patent license to a |
|  | party means to make such an agreement or commitment not to enforce a |
|  | patent against the party. |
|  |  |
|  | If you convey a covered work, knowingly relying on a patent license, |
|  | and the Corresponding Source of the work is not available for anyone |
|  | to copy, free of charge and under the terms of this License, through a |
|  | publicly available network server or other readily accessible means, |
|  | then you must either (1) cause the Corresponding Source to be so |
|  | available, or (2) arrange to deprive yourself of the benefit of the |
|  | patent license for this particular work, or (3) arrange, in a manner |
|  | consistent with the requirements of this License, to extend the patent |
|  | license to downstream recipients. "Knowingly relying" means you have |
|  | actual knowledge that, but for the patent license, your conveying the |
|  | covered work in a country, or your recipient's use of the covered work |
|  | in a country, would infringe one or more identifiable patents in that |
|  | country that you have reason to believe are valid. |
|  |  |
|  | If, pursuant to or in connection with a single transaction or |
|  | arrangement, you convey, or propagate by procuring conveyance of, a |
|  | covered work, and grant a patent license to some of the parties |
|  | receiving the covered work authorizing them to use, propagate, modify |
|  | or convey a specific copy of the covered work, then the patent license |
|  | you grant is automatically extended to all recipients of the covered |
|  | work and works based on it. |
|  |  |
|  | A patent license is "discriminatory" if it does not include within |
|  | the scope of its coverage, prohibits the exercise of, or is |
|  | conditioned on the non-exercise of one or more of the rights that are |
|  | specifically granted under this License. You may not convey a covered |
|  | work if you are a party to an arrangement with a third party that is |
|  | in the business of distributing software, under which you make payment |
|  | to the third party based on the extent of your activity of conveying |
|  | the work, and under which the third party grants, to any of the |
|  | parties who would receive the covered work from you, a discriminatory |
|  | patent license (a) in connection with copies of the covered work |
|  | conveyed by you (or copies made from those copies), or (b) primarily |
|  | for and in connection with specific products or compilations that |
|  | contain the covered work, unless you entered into that arrangement, |
|  | or that patent license was granted, prior to 28 March 2007. |
|  |  |
|  | Nothing in this License shall be construed as excluding or limiting |
|  | any implied license or other defenses to infringement that may |
|  | otherwise be available to you under applicable patent law. |
|  |  |
|  | 12. No Surrender of Others' Freedom. |
|  |  |
|  | If conditions are imposed on you (whether by court order, agreement or |
|  | otherwise) that contradict the conditions of this License, they do not |
|  | excuse you from the conditions of this License. If you cannot convey a |
|  | covered work so as to satisfy simultaneously your obligations under this |
|  | License and any other pertinent obligations, then as a consequence you may |
|  | not convey it at all. For example, if you agree to terms that obligate you |
|  | to collect a royalty for further conveying from those to whom you convey |
|  | the Program, the only way you could satisfy both those terms and this |
|  | License would be to refrain entirely from conveying the Program. |
|  |  |
|  | 13. Use with the GNU Affero General Public License. |
|  |  |
|  | Notwithstanding any other provision of this License, you have |
|  | permission to link or combine any covered work with a work licensed |
|  | under version 3 of the GNU Affero General Public License into a single |
|  | combined work, and to convey the resulting work. The terms of this |
|  | License will continue to apply to the part which is the covered work, |
|  | but the special requirements of the GNU Affero General Public License, |
|  | section 13, concerning interaction through a network will apply to the |
|  | combination as such. |
|  |  |
|  | 14. Revised Versions of this License. |
|  |  |
|  | The Free Software Foundation may publish revised and/or new versions of |
|  | the GNU General Public License from time to time. Such new versions will |
|  | be similar in spirit to the present version, but may differ in detail to |
|  | address new problems or concerns. |
|  |  |
|  | Each version is given a distinguishing version number. If the |
|  | Program specifies that a certain numbered version of the GNU General |
|  | Public License "or any later version" applies to it, you have the |
|  | option of following the terms and conditions either of that numbered |
|  | version or of any later version published by the Free Software |
|  | Foundation. If the Program does not specify a version number of the |
|  | GNU General Public License, you may choose any version ever published |
|  | by the Free Software Foundation. |
|  |  |
|  | If the Program specifies that a proxy can decide which future |
|  | versions of the GNU General Public License can be used, that proxy's |
|  | public statement of acceptance of a version permanently authorizes you |
|  | to choose that version for the Program. |
|  |  |
|  | Later license versions may give you additional or different |
|  | permissions. However, no additional obligations are imposed on any |
|  | author or copyright holder as a result of your choosing to follow a |
|  | later version. |
|  |  |
|  | 15. Disclaimer of Warranty. |
|  |  |
|  | THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY |
|  | APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT |
|  | HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY |
|  | OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, |
|  | THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR |
|  | PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM |
|  | IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF |
|  | ALL NECESSARY SERVICING, REPAIR OR CORRECTION. |
|  |  |
|  | 16. Limitation of Liability. |
|  |  |
|  | IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING |
|  | WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS |
|  | THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY |
|  | GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE |
|  | USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF |
|  | DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD |
|  | PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), |
|  | EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF |
|  | SUCH DAMAGES. |
|  |  |
|  | 17. Interpretation of Sections 15 and 16. |
|  |  |
|  | If the disclaimer of warranty and limitation of liability provided |
|  | above cannot be given local legal effect according to their terms, |
|  | reviewing courts shall apply local law that most closely approximates |
|  | an absolute waiver of all civil liability in connection with the |
|  | Program, unless a warranty or assumption of liability accompanies a |
|  | copy of the Program in return for a fee. |
|  |  |
|  | END OF TERMS AND CONDITIONS |
|  |  |
|  | How to Apply These Terms to Your New Programs |
|  |  |
|  | If you develop a new program, and you want it to be of the greatest |
|  | possible use to the public, the best way to achieve this is to make it |
|  | free software which everyone can redistribute and change under these terms. |
|  |  |
|  | To do so, attach the following notices to the program. It is safest |
|  | to attach them to the start of each source file to most effectively |
|  | state the exclusion of warranty; and each file should have at least |
|  | the "copyright" line and a pointer to where the full notice is found. |
|  |  |
|  | <one line to give the program's name and a brief idea of what it does.> |
|  | Copyright (C) <year> <name of author> |
|  |  |
|  | This program is free software: you can redistribute it and/or modify |
|  | it under the terms of the GNU General Public License as published by |
|  | the Free Software Foundation, either version 3 of the License, or |
|  | (at your option) any later version. |
|  |  |
|  | This program is distributed in the hope that it will be useful, |
|  | but WITHOUT ANY WARRANTY; without even the implied warranty of |
|  | MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the |
|  | GNU General Public License for more details. |
|  |  |
|  | You should have received a copy of the GNU General Public License |
|  | along with this program. If not, see <http://www.gnu.org/licenses/>. |
|  |  |
|  | Also add information on how to contact you by electronic and paper mail. |
|  |  |
|  | If the program does terminal interaction, make it output a short |
|  | notice like this when it starts in an interactive mode: |
|  |  |
|  | <program> Copyright (C) <year> <name of author> |
|  | This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'. |
|  | This is free software, and you are welcome to redistribute it |
|  | under certain conditions; type `show c' for details. |
|  |  |
|  | The hypothetical commands `show w' and `show c' should show the appropriate |
|  | parts of the General Public License. Of course, your program's commands |
|  | might be different; for a GUI interface, you would use an "about box". |
|  |  |
|  | You should also get your employer (if you work as a programmer) or school, |
|  | if any, to sign a "copyright disclaimer" for the program, if necessary. |
|  | For more information on this, and how to apply and follow the GNU GPL, see |
|  | <http://www.gnu.org/licenses/>. |
|  |  |
|  | The GNU General Public License does not permit incorporating your program |
|  | into proprietary programs. If your program is a subroutine library, you |
|  | may consider it more useful to permit linking proprietary applications with |
|  | the library. If this is what you want to do, use the GNU Lesser General |
|  | Public License instead of this License. But first, please read |
|  | <http://www.gnu.org/philosophy/why-not-lgpl.html>. |