Aleksi Anderson

QT Animation Editor

Technical Design Document

17.4.2023

**Table of contents**

[1 Animation Editor Overview 1](#_Toc132630174)

[1.1 Summary 1](#_Toc132630175)

[1.2 Technical goals 1](#_Toc132630176)

[1.3 Tools 1](#_Toc132630177)

[2 Architecture 2](#_Toc132630178)

[3 Software behavior 3](#_Toc132630179)

[3.1 Software loop 3](#_Toc132630180)

[4 Self-assessment 4](#_Toc132630181)

# Animation Editor Overview

## Summary

This program is meant as a simple animation editor to be used with our game engine from Project Studies 3. It lets you load images from files and turn them into an animation by setting the delay between each frame. To add a frame to the animation just load the image/images, click on the image you want to add, and press add frame.

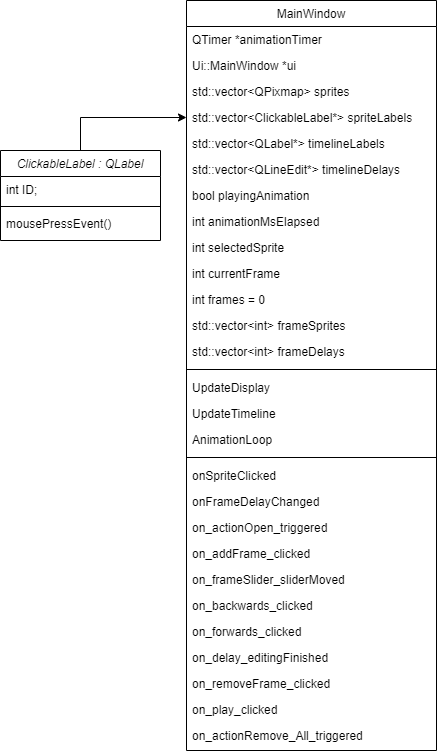
## Technical goals

The program should be able to load images, organize them into an animation, and then export that animation to a format that can easily be used in our game engine. It would also be nice to be able to slice a sprite sheet into multiple images.

## Tools

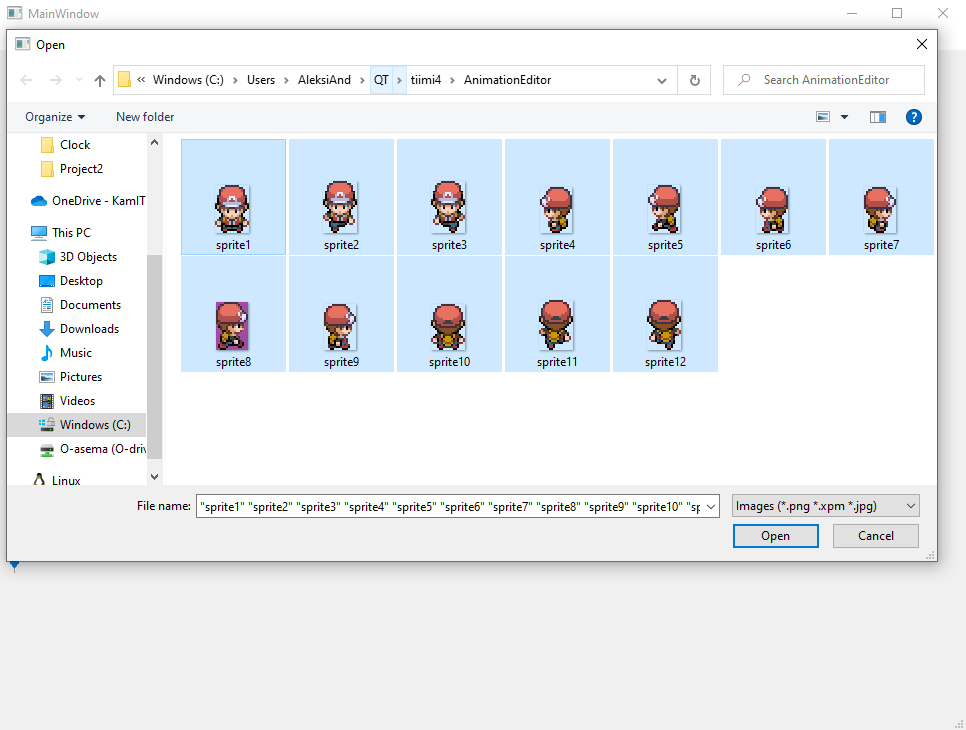
The program was made in Qt using C++.

# Architecture

Since this program is so simple, it is made entirely within the MainWindow class. The only other custom class is ClickableLabel which implements mousePressEvent to the QLabel class, this is necessary to have clickable images. The program is made in such a way that the major UI elements which need to be updated often, those being the animation preview and the timeline, have their own update functions which are then called from any onAction function that modifies data related to those UI elements. Other major UI updates such as loading images are handled in their own onAction function, since it is the only time that UI element needs to be updated. There is also a timer for the animation preview which displays the animation with the correct timing.

# Software behavior

## Software loop

There is no main loop in this program, besides of course the loops Qt uses in the background. The entire program is event driven, even the animation preview is updated by a timer event. There is an onAction function for every button and other interactable item which handles its behavior.

# Self-assessment

I would give myself a 4 for this project, because I got everything working as intended, except for exporting the animations, which I could not do here since our game engine doesn’t have the capability to load animations and we have not yet come up with a format to do that in. So technically this program is not useful in its current state.

I stayed on schedule for the whole project and completed everything but this documentation weeks before the deadline. I learned quite a bit about Qt, especially how it handles image loading and timers, as well as how to extend basic Qt widgets with custom functionality.