МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Документация

по лабораторной работе №1

Петрова Егора Андреевича

студента 2 курса, 10 группы

специальность «Прикладная информатика»

Руководитель практики:

Старший преподаватель кафедры компьютерных технологий и систем

С. С. Соловей

Минск, 2025

# Содержание

[Содержание 2](#_Toc210854191)

[Предисловие 3](#_Toc210854192)

[Часть 1 Общее описание 4](#_Toc210854193)

[1.1 Основные возможности 4](#_Toc210854194)

[1.2 Цветовые модели 4](#_Toc210854195)

[Глава 2 Руководство пользователя 6](#_Toc210854196)

[2.1 Запуск приложения 6](#_Toc210854197)

[2.2 Основные элементы интерфейса 6](#_Toc210854198)

[Глава 3 Устройство и архитектура кода приложения 8](#_Toc210854199)

[3.1 Структура проекта 8](#_Toc210854200)

[3.2 Взаимодействие компонентов 8](#_Toc210854201)

[3.3 Разбор кода 8](#_Toc210854202)

# Предисловие

Данное приложение было разработано как универсальный инструмент для работы с основными цветовыми моделями: RGB, HSV и XYZ. Оно предоставляет простое для использования средство, которое будет полезно для начинающих, изучающих теорию цвета.

Особенность "RGB-HSV-XYZ Converter" заключается в его интуитивно понятном интерфейсе, который позволяет мгновенно визуализировать изменения и наблюдать, как преобразования между разными цветовыми пространствами влияют на конечный результат. Реализованные алгоритмы преобразования обеспечивают высокую точность, а система уведомлений предупреждает пользователя о возможных неточностях и ограничениях.

Мы надеемся, что это приложение станет для вас надежным помощником в работе с цветом, будь то профессиональные задачи, образовательные цели или творческие эксперименты. Данная документация призвана помочь вам максимально эффективно использовать все возможности программы.

# Часть 1 Общее описание

## Постановка задачи

Изучить цветовые модели: RGB, CMYK, HSV, HLS, XYZ, LAB, переход от одной модели к другой, исследовать цветовой график МКО.

Создать приложение, позволяющее пользователю выбирать, а затем интерактивно менять цвет, показывая при этом его составляющие в трех моделях одновременно.

На проверку сдаются:

* + Exe-файл, который должен работать на ПК преподавателя под Windows,
  + исходный код приложения на gitHub
  + сопроводительная документация.

Основные требования к приложению:

* В интерфейсе дать возможность пользователю задавать точные цвета выбирать цвета из палитры, плавно изменять цвета.
* При изменении любой компоненты цвета все остальные представления этого цвета в двух других цветовых моделях пересчитываются автоматически.
* При «некорректных цветах» выдавать некое ненавязчивое предупреждение, что происходит обрезание-округление и т.п.

## Основные возможности

"RGB-HSV-XYZ Converter" - это приложение для работы с цветами в различных цветовых моделях.

Программа позволяет:

* Конвертировать цветовые модели: RGB, HSV и XYZ
* Визуализировать цвета: Просмотр выбранного цвета в реальном времени
* Вводить данные несколькими способами: Использование ползунков, числовых полей и диалога выбора цвета
* Автоматически синхронизировать изменения: Изменения в одной модели автоматически отражаются в других
* Проверять данные на точность: Валидация вводимых значений и уведомления о корректировках

## Цветовые модели

Цветовые модели - это математические способы описания цвета с помощью числовых значений. Различные модели предназначены для разных целей: одни удобны для технической реализации (RGB), другие - для интуитивного восприятия человеком (HSV), третьи - для научных исследований (XYZ).

* + 1. **Модель RGB**

**Тип модели**: Аддитивная (цвет формируется сложением световых потоков.

**Компоненты**:

* R (Красный) - 0-255
* G (Зеленый) - 0-255
* B (Синий) - 0-255

Применение: Компьютерные мониторы, телевизоры, цифровые дисплеи

**Преимущества**:

* Прямое соответствие аппаратному обеспечению
* Простота реализации в электронных устройствах
* Широкое распространение

Особенности в приложении:

* Целочисленные значения от 0 до 255
* Плавная синхронизация между ползунками и числовыми полями
* Немедленное визуальное отображение изменений
  + 1. **Модель HSV**

**Тип модели**: Цилиндрическая (основана на человеческом восприятии)

**Компоненты**:

* H (Оттенок) - 0°-360° (цветовой круг)
* S (Насыщенность) - 0%-100% (интенсивность цвета)
* V (Яркость) - 0%-100% (уровень освещенности)

Применение: Графические редакторы, дизайн, подбор цветовых схем.

**Преимущества**:

* Интуитивно понятна для человека
* Удобна для подбора гармоничных цветов
* Позволяет легко регулировать отдельные аспекты цвета

Особенности в приложении:

* Дробные значения для большей точности
* Циклическая природа оттенка (0° и 360° - одинаковые значения)
* Визуально логичная организация параметров
  + 1. **Модель XYZ**

**Тип модели**: Научная, аппаратно-независимая

**Компоненты**:

* X - 0.0-95.05
* Y - 0.0-100.0 (яркость)
* Z - 0.0-108.9

Применение: Научные исследования, колориметрия, точные цветовые измерения.

**Преимущества**:

* Аппаратная независимость
* Охватывает все цвета, видимые человеческим глазом
* Основа для других цветовых пространств (LAB, LUV)

Особенности в приложении:

* Высокая точность (3 знака после запятой)
* Сложные преобразования с RGB
* Автоматическая проверка на достижимость цветов

# Глава 2 Руководство пользователя

## 2.1 Запуск приложения

* Через exe-файл на любом компьютере под Windows
* Запуск через QtCreator

## 2.2 Основные элементы интерфейса

Область просмотра цвета:

* Расположена в левой части окна
* Отображает текущий выбранный цвет
* Кнопка "Choose Color" открывает стандартный диалог выбора цвета

Панель RGB:

* Три компонента: Red (R), Green (G), Blue (B)
* Для каждого компонента:
  + Спин-бокс для точного ввода значения (0-255)
  + Ползунок для визуального изменения значения

Панель HSV:

* Три компонента: Hue (H), Saturation (S), Value (V)
* Для каждого компонента:
  + Спин-бокс с поддержкой дробных значений
  + Ползунок для визуального изменения

Панель XYZ:

* Три компонента: X, Y, Z
* Для каждого компонента:
  + Спин-бокс с поддержкой дробных значений (3 знака после запятой)
  + Ползунок для визуального изменения

Строка состояния:

* Отображает информационные сообщения:
  + Уведомления о корректировках значений
  + Сообщения об ошибках валидации
  1. **Работа с приложением**

**Изменение цвета:**

* **Через панели управления**:
  + Измените значение в любой из панелей (RGB, HSV или XYZ)
  + Используйте спин-боксы для точного ввода
  + Используйте ползунки для быстрой регулировки
* **Через диалог выбора цвета**:
* Нажмите кнопку "Choose Color"
* Выберите цвет в появившемся диалоговом окне
* Нажмите "OK" для применения

**Особенности работы:**

* При изменении значения в одной цветовой модели автоматически пересчитываются значения в других моделях
* При вводе значений вне допустимого диапазона они автоматически ограничиваются
* При значительных округлениях во время конвертаций появляются уведомления

# Глава 3 Устройство и архитектура кода приложения

## 3.1 Структура проекта

Приложение построено по классической схеме Model-View-Controller (MVC) с использованием фреймворка Qt:

RGB-HSV-XYZ Converter/

├── main.cpp # Точка входа

├── mainwindow.h/cpp # View + Controller

└── colormodel.h/cpp # Model

## 3.2 Взаимодействие компонентов

Пользовательский интерфейс (MainWindow)

↑

| Сигналы/Слоты Qt

↓

Цветовая модель (ColorModel)

↑

| Математические преобразования

↓

Цветовые пространства (RGB/HSV/XYZ)

## 3.3 Разбор кода

* + 1. main.cpp:

## #include "mainwindow.h"

## #include <QApplication>

## int main(int argc, char \*argv[])

## {

## QApplication a(argc, argv); // Инициализация Qt приложения

## MainWindow w; // Создание главного окна

## w.show(); // Отображение окна

## return a.exec(); // Запуск цикла обработки событий

* + 1. **Цветовая модель ColorModel:**

**Структуры данных:**

struct RGB{

int r, g, b;

RGB(int red = 0, int green = 0, int blue = 0) : r(red), g(green), b(blue) {}

};

struct HSV{

double h, s, v;

HSV(double hue = 0, double saturation = 0, double value = 0) : h(hue), s(saturation), v(value) {}

};

struct XYZ{

double x, y, z;

XYZ(double xVal = 0, double yVal = 0, double zVal = 0) : x(xVal), y(yVal), z(zVal) {}

};

**Конструктор модели:**

ColorModel::ColorModel(QObject \*parent) :

QObject(parent),

m\_rgb(255, 255, 255), // Белый цвет по умолчанию

m\_hsv(0, 0, 100), // HSV представление белого

m\_xyz(95.05, 100.0, 108.9), // XYZ представление белого

m\_isValid(true) // Флаг валидности

{}

**Алгоритмы преобразования:**

HSV ColorModel::**rgbToHsv**(const RGB& rgb) const

{

double red = rgb.r / 255.0;

double green = rgb.g / 255.0;

double blue = rgb.b / 255.0;

double max = qMax(red, qMax(green, blue));

double min = qMin(red, qMin(green, blue));

double delta = max - min;

HSV hsv;

hsv.v = max \* 100.0; // Value (яркость)

if (max < 0.0001) {

hsv.s = 0; // Насыщенность 0 для черного

} else {

hsv.s = (delta / max) \* 100.0; // Насыщенность

}

// Вычисление оттенка

if (delta < 0.0001) {

hsv.h = 0; // Ахроматический цвет

} else if (red >= max) {

hsv.h = 60.0 \* fmod(((green - blue) / delta), 6.0);

} else if (green >= max) {

hsv.h = 60.0 \* (((blue - red) / delta) + 2.0);

} else {

hsv.h = 60.0 \* (((red - green) / delta) + 4.0);

}

if(hsv.h < 0) hsv.h += 360.0; // Коррекция отрицательных значений

return hsv;

}

RGB ColorModel::**hsvToRgb**(const HSV& hsv) const

{

double h = clamp(hsv.h, 0, 360);

double s = clamp(hsv.s, 0, 100) / 100.0;

double v = clamp(hsv.v, 0, 100) / 100.0;

double c = v \* s; // Хрома

double x = c \* (1 - fabs(fmod(h / 60.0, 2) - 1));

double m = v - c; // Минимальное значение

double r, g, b;

// Разбиение цветового круга на 6 секторов

if (h < 60) {

r = c; g = x; b = 0;

} else if (h < 120) {

r = x; g = c; b = 0;

} else if (h < 180) {

r = 0; g = c; b = x;

} else if (h < 240) {

r = 0; g = x; b = c;

} else if (h < 300) {

r = x; g = 0; b = c;

} else {

r = c; g = 0; b = x;

}

return RGB(

clamp((r + m) \* 255, 0, 255),

clamp((g + m) \* 255, 0, 255),

clamp((b + m) \* 255, 0, 255)

);

}

XYZ ColorModel::**rgbToXyz**(const RGB& rgb) const

{

double red = rgb.r / 255.0;

double green = rgb.g / 255.0;

double blue = rgb.b / 255.0;

// Гамма-коррекция

red = ((red >= 0.04045) ? pow((red + 0.055) / 1.055, 2.4) : red / 12.92) \* 100;

green = ((green >= 0.04045) ? pow((green + 0.055) / 1.055, 2.4) : green / 12.92) \* 100;

blue = ((blue >= 0.04045) ? pow((blue + 0.055) / 1.055, 2.4) : blue / 12.92) \* 100;

// Матричное преобразование

double x = red \* 0.412453 + green \* 0.35758 + blue \* 0.180423;

double y = red \* 0.212671 + green \* 0.71516 + blue \* 0.072169;

double z = red \* 0.019334 + green \* 0.119193 + blue \* 0.950227;

return XYZ(x, y, z);

}

**Механизм обновления:**

void ColorModel::update()

{

emit colorChanged(color()); // Сигнал об изменении цвета

emit rgbChanged(m\_rgb); // Сигнал об изменении RGB

emit hsvChanged(m\_hsv); // Сигнал об изменении HSV

emit xyzChanged(m\_xyz); // Сигнал об изменении XYZ

}

* + 1. **Главное окно (MainWindow)**

MainWindow

├── CentralWidget

│ ├── GridLayout

│ │ ├── PreviewGroup (0,0)

│ │ ├── RGBGroup (0,1)

│ │ ├── HSVGroup (1,1)

│ │ └── XYZGroup (2,1)

└── StatusBar