KALK

[Selezionare la data]

Progetto del corso di Programmazione ad Oggetti a.a 2017/2018

CHINELLO ANDREA 1143621

Kalk

PROGETTO DEL CORSO DI PROGRAMMAZIONE AD OGGETTI A.A 2017/2018

Il progetto è stato svolto in coppia con GIACOMO PONTARA 1126744

prefazione

SCOPO DEL PROGETTO

Lo scopo del progetto è quello di realizzare una calcolatrice che sia in grado di gestire ed effettuare delle operazioni su Colori, su Palette e su PixelArt, ovvero un'immagine rappresentata tramite singoli pixel di colore.

STRUMENTI UTILIZZATI

Il progetto presenta due diverse versioni.

La prima versione è il progetto Kalk in C++. Questa versione dispone di un'interfaccia grafica realizzata utilizzando la libreria Qt e il framework ad essa associato nella versione 5.5.1, messa a disposizione nella macchina virtuale fornita durante il corso.

La seconda versione è stata ideata in Java e può essere usata solo tramite la Linea di comando.

NUMERO DI ORE RICHIESTE PER LA REALIZZAZIONE DEL PROGETTO

- identificazione della tipologia di calcolatrice da creare, analisi dei requisiti: 5 ore (comuni)
- progettazione modello e ricerca degli algoritmi da impiegare: 10 ore (comuni)
- progettazione GUI: 3 ore (2 comuni, 1 di Andrea)
- sviluppo grafica: 25 ore (10 comuni, 15 di Andrea)
- sviluppo modello: 25 ore (10 comuni, 15 di Giacomo)
- test: 5 ore (comuni)

SUDDIVISIONE RESPONSABILITA' PROGETTUALI

Andrea si è occupato nello specifico della creazione dei Widget per la rappresentazione a schermo dei tipi di calcolo, creando custom Widgets che rispondano alle esigenze di display di Kalk. Giacomo si è occupato invece di scrivere il codice delle classi del modello, riadattando gli algoritmi comunemente usati nell'ambito della Color Theory.

FUNZIONALITÀ PRINCIPALI

Kalk è un'applicazione volta a definire operazioni su Colori e su strutture basate su diverse tipologie di aggregazione di essi, come la Palette, in cui selezionare e modificare specifici colori, la PixelArt, ovvero un'immagine rappresentata tramite pixel colorati (per rappresentare semplici figure o loghi) oppure un'Immagine Speculare, ovvero una particolare PixelArt che ha come caratteristica identificativa il fatto di essere orizzontalmente o verticalmente o entrambi speculare rispetto al suo centro.

Un Colore può essere rappresentato secondo vari modelli. Il progetto utilizza i modelli RGB ed HSV.

Ogni Colore RGB è rappresentato da tre coordinate (Red , Green , Blue) che indicano ciascuna il corrispondente livello dei colori base del modello (Rosso, Verde, Blu), da 0 a 255. Lo spazio di rappresentazione così definito permette di rappresentare 255^3 colori, per un totale di 16.581.375 di essi.

Il modello HSV (Hue=Tonalità/Tinta , Saturation=Saturazione e Value=Valore) definisce i colori secondo un angolo (la Tonalità/Tinta, con Rosso che vale 0°, Verde 120° e Blu 240°) e due valori compresi fra 0 e 1, rappresentabili pertanto tramite una percentuale (la Saturazione, che indica l'intensità della Tinta, Valore che indica la luminosità).

Le operazioni permesse da Kalk usano uno o più colori dati come input tramite le coordinate RGB, rappresentando a schermo i risultati ottenuti sia come coordinate, sia come display effettivo del Colore o dei Colori (quando si tratta di operazioni su strutture aggregate) ottenuti.

Si possono compiere operazioni Unarie come la Desaturazione, la ricerca di un complementare, ricavare i cinque colori che compongono L'Analogo, ottenere schemi quali Tetrade e Triade, con prevista la conversione a schermo in notazione esadecimale, utile in ambiti come il web design in cui tale notazione è molto diffusa.

Sono previste anche operazioni Binarie come la somma, la sottrazione, la moltiplicazione e la divisione fra colori, i confronti in termini di luminosità.

MODELLO

Color è la classe fondante del progetto. Rappresenta il modello logico di rappresentazione di un Colore tramite le sue coordinate RGB.

Ha come campi dati interi r, g, b. E' presente pure il campo dati a, che rappresenta l'opacity. Tale valore non è stato usato ai fini del progetto, ma rappresenta la possibilità di fare operazioni sui Color anche in termine di opacità. Tali campi hanno ciascuno a disposizione un metodo setter e un metodo getter.

Mette a disposizione i seguenti metodi:

Complementare, calcola il colore opposto a quello di partenza, ovvero il colore che ha valore H (Hue, ovvero Tonalità) opposto rispetto a quello di partenza. Ne restituisce il valore Color (ovvero la notazione RGB).

Desat, calcola il colore Desaturato, ovvero in una tonalità tendente al grigio.

Analogo, calcola uno schema di 5 colori equidistanti in termini di Tonalità (tipicamente con uno spostamento di 30° in HSV).

Triade, calcola uno schema di 3 colori affine all'analogo, ma con variazione di 120°.

Tetrade, calcola uno schema di 4 colori affine ad Analogo e Triade, ma con variazione di 90°.

Wavelength, permette di convertire una lunghezza d'onda in nanometri presa da uno spettro di luce in un Color, rappresentando la componente colorata di tale lunghezza d'onda. Riprende un algoritmo di conversione che permette di dividere lo spettro visibile (fra i 350 e i 780 nm) in fasce di colore, in modo affine alla rifrazione.

Ridefinisce i seguenti operatori:

Somma (operator +), che somma le coordinate RGB di due colori ottenendo un terzo colore che è la sovrapposizione grafica del secondo colore rispetto al primo.

Sottrazione (operator -), che dati due colori ne crea uno nuovo sottraendo le coordinate RGB del secondo al primo.

Moltiplicazione (operator *) sia fra due colori moltiplicando le rispettive coordinate per ottenere le coordinate del colore risultato, sia fra un color e un numero intero n, moltiplicando n volte le coordinate del color.

Divisione (operator /) sia fra due colori dividendo le rispettive coordinate (se le coordinate del secondo sono !=0) per ottenere le coordinate del colore risultato, oppure dividendo per un intero n le coordinate del colore di partenza (se n !=0). Qualora si verificasse una divisione per 0 è previsto che l'utente sia allertato del tentativo di operazione logica sbagliato.

Maggiore (operator >) che fra due colori restituisce il valore true se il primo ha luminosità (ovvero la Value del modello HSV) maggiore del secondo, altrimenti false.

Minore (operator <) che svolge funzione opposta all'operatore Maggiore.

ColorGroup classe base astratta che definisce il concetto di raggruppamento colorato.

Su di esso sono previste in maniera astratta pura le operazioni di Somma (su tutti i colori presenti nel gruppo), Complementare (su tutti i colori presenti nel gruppo), Trova e Rimuovi (che dato un Colore lo trova e elimina dal gruppo), Desaturazione (che desatura tutti i colori presenti nel gruppo), Opacity che sfrutta la feature (non implementata nella grafica del progetto) di modificare l'opacità di tutti gli elementi del gruppo. Tutte queste operazioni sono overridate per ciascun ColorGroup.

Palette è classe figlia di ColorGroup e rappresenta una tavolozza di colori. Nella View si è deciso di limitare le sue dimensioni a 5 colori.

Palette ha come campi dati una vector di Color e un campo dati intero che ne rappresenta le dimensioni.

Oltre a fornire override dei metodi della classe ColorGroup, fornisce i metodi:

Fill trasforma tutti i colori della palette nel colore scelto.

ReturnIndexMax trova il massimo colore per brillantezza nella palette e ne ritorna l'indice.

ReturnIndexMin trova il minimo colore per brillantezza nella palette e ne ritorna l'indice.

Rimuovi, che rimuove dalla palette il colore con indice passato.

Sostituisci, che cambia il colore sulla palette con indice passato con un nuovo colore passato.

Inserisci, che inserisce il colore passato in ultima posizione della vector.

PixelArt è classe figlia di ColorGroup e rappresenta un'immagine definita da una griglia di Colori. Nella view si è deciso di limitare le sue dimensioni a una griglia 8x8.

PixelArt ha come campi dati il numero intero di righe e di colonne della sua griglia, nonché una matrice di Color di tali dimensioni che ospita la sua rappresentazione.

Oltre a fornire override dei metodi della sua classe padre ColorGroup, fornisce i metodi:

Add colore date delle coordinate sulla griglia sostituisce il colore lì presente con il colore passato. E' un metodo virtuale che trova override nella classe figlia ImgSpeculare.

scambiaVert scambia la metà superiore della griglia con la metà inferiore, in maniera speculare rispetto al centro (o alla riga centrale se il numero di righe è dispari e la riga centrale diventa centro di simmetria).

scambiaOrizz scambia la metà sinistra della griglia con la metà destra, in maniera speculare rispetto al centro (o alla colonna centrale se il numero di righe è dispari e la colonna centrale diventa centro di simmetria).

isSimmetricoX ritorna true se la griglia è simmetrica rispetto al centro, ovvero quando ciascun colore della metà superiore della griglia corrisponde a un altro colore uguale nella metà inferiore in coordinate corrispondenti

isSimmetricoY ritorna true se la griglia è simmetrica rispetto al centro, ovvero quando ciascun colore della metà sinistra della griglia corrisponde a un altro colore uguale nella metà destra in coordinate corrispondenti

ImgSpeculare è classe figlia di PixelArt, si specifica nel compiere operazioni di inserimento nella griglia di PixelArt, che mantengano però la proprietà di specularità (o orizzontale, o verticale, o entrambe se l'immagine divisa in quadranti ha i 4 quadranti speculari rispetto al centro verticale e orizzontale).

ImgSpeculare ha come campi dati booleani Vert e Oriz, che ne rappresentano le proprietà di simmetria verticale e orizzontale.

Fornisce override del metodo addColore di PixelArt. Tale metodo viene invocato anche sui colori di coordinate corrispondenti al colore che si intende modificare, in modo da svolgere la sostituzione preservando le proprietà di specularità (orizzontale se orizzontale, verticale se verticale, oppure entrambe).

flipX metodo che prende la metà destra della griglia e per ciascun elemento vi sostituisce il corrispettivo simmetrico sinistro. Così facendo l'imgSpeculare acquisisce proprietà di specularità verticale.

flipY metodo che prende la metà superiore della griglia e per ciascun elemento vi sostituisce il corrispettivo simmetrico inferiore. Così facendo l'imgSpeculare acquisisce proprietà di specularità orizzontale.

VIEW

MainWindow è la classe esterna della view. Contiene il QTabWidget che divide la GUI dell'applicazione in 3 tab distinte: Colori, Palette, PixelArt. All'interno delle 3 tab vengono richiamate delle sottoclassi di QWidget ridefinite. Esse contengono la rappresentazione grafica delle tre tab, comprensive di altri widget ridefiniti.

colorDisplay è il widget ridefinito da QWidget, gestisce interamente la rappresentazione a schermo di un Color e le sue operazioni unarie.

TabColori è il QWidget che contiene una QGridLayout, con a sua volta all'interno due widget colorDisplay e i bottoni delle operazioni binarie su Colori.

TabPalette contiene un Widget chiamato PaletteWidget, che contiene a sua volta il widget DisplayPalette che ospita cinque Widget ColorCard, ciascuna ColorCard mostra il colore assegnato ad essa, più un tasto modifica e un tasto rimuovi. Il tasto modifica permette il display del colore nella parte destra e permette di modificarlo. Il tasto rimuovi invece rende bianco il colore. Le operazioni su tutta la palette sono gestite tramite pulsanti presenti all'interno di PaletteWidget nella sua parte inferiore.

TabPixelArt contiene una QTableWidget usata per rappresentare la griglia. Contiene i tasti nella sua parte inferiore sinistra e un widget ColorDisplay nella parte destra. A click su una singola cella il colore viene portato sul ColorDisplay destro che permette modifiche e inserimento nella griglia. In questa tab è previsto anche il Tasto "abilita speculare" che modifica il comportamento della tab specializzando la PixelArt in immagine speculare. A ogni operazione in modalità speculare viene fatto un controllo dinamico di tipo, per vedere se mantiene la proprietà di specularità. Tale proprietà può essere persa in caso di disattivazione della modalità speculare e modifica in modalità PixelArt. In tal caso un tentativo di riattivazione di modalità speculare causa un warning se le proprietà di specularità della PixelArt non sono rispettate nella creazione di immagine speculare.

GUIDA ALL'UTILIZZO

Nella tab **Colori** si può creare un colore inserendo le 3 coordinate RGB (fra 0 e 255). Appare a schermo anche la corrispondente conversione in notazione esadecimale del colore definito.

Complementare e Desat svolgono la corrispondente funzione e displayano il colore creato. Analogo Triade e Tetrade generano 5, 3 e 4 colori in un'armonia e la posizionano negli spazi corrispondenti.

Wavelength genera il colore corrispondente alla lunghezza d'onda generata.

Le operazioni binarie nella parte bassa della tab abilitano una seconda scheda di selezione colore che permette di costruire un secondo colore da usare come secondo operando. L'operazione di divisione fra colori ha implementato un controllo di divisione per zero, che manda un warning in caso di tentativo di divisione con un secondo operando che ha una delle 3 coordinate rgb = 0.

Nella tab Palette i 5 colori presenti in alto possono essere modificati (tramite il menù di selezione e costruzione colore di destra), oppure rimossi. In basso sono presenti le operazioni eseguibili su tutto il colorgroup. L'operazione trova e rimuovi utilizza il menù destro per definire il colore da rimuovere dalla palette. Le operazioni che hanno come risultato un colore ne fanno display nel menù destro (trova > fa display del colore più brillante della palette, trova < di quello meno brillante).

La tab PixelArt ha in alto a sinistra la griglia 8x8 che fa display della matrice. Si possono modificare le celle tramite il menù di selezione colore destro. Il comando clean ripulisce la griglia e la riporta alla situazione di partenza. Desat complem scambiaV e scambiaH creano sulla griglia l'effetto descritto dai corrispondenti metodi e hanno display del risultato direttamente sulla griglia. Add e trova e rimuovi prendono in input un colore costruito tramite il menù di creazione colore destro. Il pulsante "abilita speculare" permette di abilitare le funzioni di immagine speculare se la pixelart attuale è speculare orizzontalmente o verticalmente (o entrambe). In modalità immagine speculare le operazioni svolte mantengono la proprietà della specularità. FlipX e FlipY permettono di vedere a schermo i corrispondenti metodi che rendono la metà superiore e quella destra uguale alla metà inferiore e sinistra. Se si disabilita la modalità speculare per riabilitarla bisogna sincerarsi di avere davanti una PixelArt speculare, altrimenti appare un warning a schermo che indica la mancanza di specularità e impedisce di entrare in modalità speculare.

MATERIALE CONSEGNATO

La cartella principale contiene 3 elementi :

1) Cartella progetto c++
2) Cartella progetto java
3) Relazione

La cartella progetto c++ contiene tutti i file .h, tutti i file .cpp ed il file .pro

Il File .pro è indispensabile per creare il MakeFile

Utilizzare qmake kalk.pro per la generazione del MakeFile corretto

La cartella del progetto di Java contiene tutti i file . java e il file use.java che fornisce un main d'esempio del modello java. Per eseguirlo usare i comandi **javac use.java** e poi **java use**