Animacje poklatkowe

Autorzy: Maciej Walczak, Przemysław Nowak

Opis projektu

Projekt polegał na napisaniu programu, który pozwalał na tworzenie pojedynczej klatki z animacji. Umożliwia on rysowanie różnych obiektów wektorowych, wybranie tła, regulacje jasności tła oraz wybranie koloru kształtów. Program tworzy jako wynik plik tekstowy, w którym zapisane są instrukcje rysujące.

Założenia wstępne przyjęte w realizacji projektu

Program powinien pozwolić utworzenie pojedynczej klatki z całej animacji. Poprzez podwójne kliknięcie na ekranie do rysowania użytkownik wybiera lewy górny róg prostokąta w którym będzie rysowany podany kształt. Utworzoną klatkę program po kliknięciu przycisku „Zapisz" albo „Kolejna klatka" będzie zapisywał w pliku tekstowym.

Analiza projektu

Specyfikacja danych wejściowych

Nasz projekt nie posiada danych wejściowych, które użytkownik może wykorzystać podczas jego działania. Jedyną rzeczą są ikony kształtów, które pełnią rolę graficzną, przyjazną dla oka.

Opis oczekiwanych danych wyjściowych

Danymi wyjściowymi jest plik tekstowy z informacją o ilości klatek oraz o wszystkich kształtach w danej klatce.

Zdefiniowanie struktur danych

Każdy kształt jest składową klasy Shape, przechowuje ona informacje o wierzchołkach, rodzaju kształtu, czy jest wypełniony czy nie, kolorze linii oraz kolorze wypełnienia.

Specyfikacja interfejsu użytkownika

Użytkownik ma możliwość wyboru jednego z sześciu kształtów (okrąg, elipsa, trójkąt, kwadrat, krzywa i łamana) za pomocą sześciu środkowych przycisków.

Przy pomocy palet u dołu wybiera kolory odpowiednio linii i wypełnienia. Jeśli okrąg, elipsa, trójkąt lub kwadrat mają być z wypełnieniem użytkownik zaznacza to przy pomocy przycisku znajdującego się najniżej.

Przycisk Browse służy do wybrania zdjęcia, które ma posłużyć za tło, a suwak pod nim zmienia jasność tego tła.

Po wyborze kształtu użytkownik ma możliwość narysowania go na ekranie po prawej stronie. Należy kliknąć 2 razy LPM żeby wybrać lewy górny róg prostokąta, w którym będzie rysowany wybrany kształt, a następnie kliknąć ponownie 2 razy LPM, żeby wybrać prawy dolny róg. Wybrany kształt będzie narysowany w danym obszarze.

W interfejsie znajdują się również przyciski „Kolejna Klatka" i „Zapisz", które służą do zapisania właściwości kształtów znajdujących się aktualnie na panelu do rysowania. W przypadku tego drugiego dochodzi jedynie do zapisu w pliku tekstowym, zaś w przypadku pierwszego oprócz zapisu następuje wyczyszczenie panelu do rysowania (tło zostaje takie samo).

Dodatkowo program jest wyposażony w moduł wyświetlania animacji który by uruchomić należy wybrać opcję wyświetl, która otwiera nowe okno do odtwarzania animacji.

Wyodrębnienie i zdefiniowanie zadań

Zadania można podzielić następująco:

-Wygenerowanie okna aplikacji

-Wybór tła oraz jego jasności

-Wybór kształtu oraz jego koloru i wypełnienia

-Rysowanie pojedynczej klatki przez użytkownika

-Zapis informacji o klatce do pliku i przejście do kolejnej

-Wyświetlanie zapisanej animacji

Decyzja o wyborze narzędzi programistycznych

Do naszego projektu zdecydowaliśmy się użyć biblioteki wxWidgets, ponieważ ta bibliotek wspiera bardzo wygodnie w budowaniu aplikacji wxFormBuilder oraz posiada przyjazny dla programisty interfejs. Wykorzystaliśmy środowisko Visual Studio, ponieważ stwierdziliśmy, że wystarczająco dobrze poznaliśmy jego możliwości przez ostatni semestr.

Podział pracy i analiza czasowa

Większość projektu zrobiliśmy wspólnie. Na początku stworzyliśmy interfejs w wxFormBuilder, gdzie dodaliśmy wszystkie potrzebne według nas przyciski i suwaki. Następnie zajęliśmy się implementacją funkcji rysujących dane kształty oraz wczytujących i modyfikujących tło naszej klatki. Na samym końcu skupiliśmy się na poprawnym zapisie do pliku.

Opracowanie niezbędnych algorytmów

Algorytm użyty przy rysowaniu poszczególnych kształtów polega na pobraniu dwóch punktów od użytkownika w celu wyznaczenia prostokąta (jak w Paincie) w którym rysowany będzie wybrany

kształt. Następnie jest to wykorzystane w funkcji rysującej wektory, aby nie rysować tylko pojedynczych kształtów podczas jednej klatki.

Algorytm służący do zapisu klatki do pliku polega na napisaniu najpierw ilości klatek, a następnie w każdej linijce opisania danej klatki. Algorytm ten podaje rodzaj kształtu, wierzchołki jego prostokąta, w który jest wpisany oraz kolory wypełnienia i linii.

Wykorzystaliśmy również algorytm do zmiany jasności tła, który działa niezawodnie dla każdego wczytanego obrazka.

Ponadto udało nam się stworzyć algorytm odczytujący podaną klatkę z pliku tekstowego.

Kodowanie

Opis klas

- **MyApp** jest to klasa niezbędna do uruchomienia programu
- **GUI** jest to klasa wygenerowana przez program wxFormBuilder, znajdują się w niej wszystkie elementy interfejsu naszego programu
- **GUIMyFrame** jest to klasa odpowiedzialna za wszystkie metody pozwalające na działanie programu
- **Shape** jest to klasa przechowująca wszystkie informacje o kształcie, oraz metody pozwalające dostać się do tych informacji
- **GUIAnimationFrame** jest to klasa obsługująca wczytywanie animacji z pliku txt a następnie jej wyświetlania.

Opis zmiennych

Std::vector<Shape> m_shapes- wektor przechowujący całą klatkę, czyli wszystkie kształty które zostały utworzone.

Shape m_actual_shape{0} – zmienna przechowująca aktualnie rysowany kształt.

Int m_drawing_flag{false} – flaga przechowująca informacje o tym jaki kształt jest rysowany.

Bool m_first_click_flag{true}- flaga przechowująca informacje czy użytkownik wybrał pierwszy punkt na panelu do rysowania.

Bool m_fill {false} –zmienna przechowująca informacje czy guzik od wypełnienia kształtu jest wciśnięty czy też nie.

Std::shared_ptr<wxImage> m_background_image_org {nullptr} – wskaźnik do oryginalnego zdjęcia używanego jako tło.

Std::shared_ptr<wxImage> m_background_image_dis {nullptr} – wskaźnik do aktualnegoo zdjęcia. używanego jako tło

wxBitmap m_background_bitmap {wxNullBitmap} – zawiera bitmapę tła.

Opis funkcji

Void paint_on_wxpanale()-funkcja odpowiadająca za rysowanie naszych kształtów po panelu

Void draw_vector_with_dc(std::shared_ptr<wxClientDC> DC)- funkcja rysująca wektory kształtów.

Void save_vector_to_file()-funkcja zapisująca wektor do pliku tekstowego.

Void reset_bitmap_buttons()-funkcja zmieniająca wygląd przycisków od kształtów.

Void correct_brightness(wxImage& image_to_change)-funkcja zmieniająca jasność podanego obrazka.

Void load_frame(int number_of_frame) – funkcja wczytuje klatkę animacji o danym numerze.

Void paint_frame() - funkcja rysuje wczytaną do klasy klatkę.

Testowanie

Testy aplikacji prowadzone były na bieżąco, początkowo zależało nam na poprawnym rysowaniu kształtów, następnie zajęliśmy się rysowaniem kilku kształtów naraz oraz przypisaniem do nich koloru, końcowo zapisem klatki do pliku. Po wykonaniu opcji podstawowych spróbowaliśmy wyposażyć program w moduł pozwalający odtworzyć animację z zadaną prędkością.

Wdrożenie, raport i wnioski

W programie udało się wykonać wszystkie wymagania podstawowe, możliwy jest wybór tła oraz jego jasności. Możliwym również jest rysowanie konkretnego kształtu z wypełnieniem lub bez oraz zapisanie aktualnej klatki i przejście do następnej. Udało się wdrożyć większą część zadań rozszerzonych jak odtwarzanie animacji, cofanie rysowania i kolory linii oraz wypełnienia.