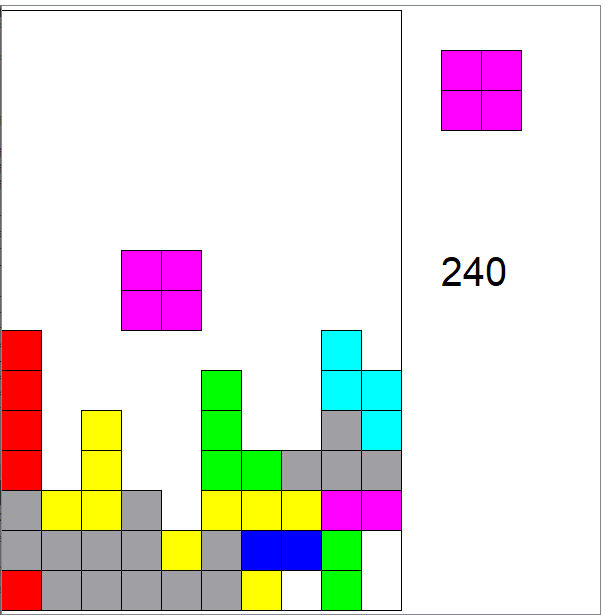
**Programowanie wieloplatformowe: Tetris (C++, Qt)**

*Bartłomiej Jakubowski*



**Klasy**

- MyPiece

- PiecesHolder

- main

**Szczegółowy opis**

MyPiece

W tej klasie zawarte zostały wszelkie informacje dotyczące klocków, ich rodzajów, możliwych pozycji oraz współrzędnych, jakie te klocki zajmują na polu gry. Na samym początku niezbędne było określenie elementów, z którymi będziemy mieli do czynienia. Klasycznymi klockami dla Tetrisa są: Z, odwrócone Z, linia, T, kwadrat, L oraz odwrócone L. Każdy z klocków składa się z 4 elementów, więc można go opisać przy pomocy współrzędnych na osi (x,y). Na samym początku, już w konstruktorze ustalamy więc „ciało” pojedynczego klocka, w zależności od losowej wartości z zakresu 0-6. Ciało takie jest reprezentowane przez tablicę dwuwymiarową 2x4 gdzie każdemu z czterech klocków przypisujemy współrzędną x oraz współrzędną y. Celem umożliwienia późniejszego prostego obrotu macierzy wartości rozpoczynają się od -1.

W tej klasie zostały również zawarte metody, zwracające maksymalne wysunięcie klocka na osiach (zarówno w jedną jak i drugą stronę) oraz takie pozwalające zwrócić nam jego pozycję po obrocie o 90 stopni. Wszystkie zwracane wartości operują na współrzędnych wewnętrznych klocka, tj. nie opisanych jeszcze na scenie, co pozwala nam zachować klocek zawsze w tej samej postaci, i modyfikować jedynie jego reprezentację na scenie.

PiecesHolder

Klasa jest w rzeczywistości całym polem gry, które zawieramy na scenie. Klasa ta definiuje sposób działania gry, wszystkie potrzebne nam zmienne oraz zmienia wartości wyświetlane na ekranie. Wszystkie wyświetlone obiekty dziedziczą po klasie QGraphicsItem (samo pole gry to QGraphicsRectItem, natomiast wyświetlamy również np. pole tekstowe typu QGraphicsSimpleTextItem).

Poszczególne klocki nie są jedną figurą, a zbiorem czterech kwadratów typu QGraphicsRectItem o rozmiarze zależnym od szerokości okna. Kolor takich kwadratów zależny jest od aktualnie aktywnej figury. Każdy kwadrat rysowany jest tylko raz, następnie jest przemieszczany o jego szerokość pomnożoną przez odpowiednią wartość (w zależności od żądanego przesunięcia).

Większość zachodzących w tle procesów wyzwalane jest przez sygnał timeout z uruchamianego już w konstruktorze QTimera, odliczającego czas do wartości odpowiednio pomniejszonej o nasz wynik (zaczynamy od 500ms, maksymalnie schodzimy do 200ms).

Akcje podzielone zostały na dwie kategorie: te, wykonywane tylko raz (zależne od wartości bool initFinished), oraz te, wykonywane wielokrotnie. Mamy więc tutaj przesunięcia klocków w osi x, opuszczanie klocka w osi y (uruchamiane przez timer lub klawisz) oraz sprawdzanie warunku pełnej linii.

main

Okno główne mojego programu to nic innego jak klasa w której tworzę scenę o stałym rozmiarze, wymuszam go na użytkowniku, umieszczam na scenie instancję wcześniej opisanej klasy PiecesHolder (do której przekazuję przez setFocus() wszystkie sygnały generowane np. przez wciskanie klawiszy), a samą scenę umieszczam w widoku QGraphicsView i wyświetlam ją metodą show().

**Podsumowanie**

W projekcie zostały wykorzystane elementy biblioteki Qt takie jak QGraphicsItem, QTimer, sygnał generowany przez ten timer, utworzony został slot z którym tym sygnał został połączony. Sama klasa główna wykorzystuje Graphics View Framework celem prezentacji obiektów graficznych.