Reversi

Instrukcja dla programisty

Katarzyna Miernikiewicz

31 stycznia 2018

1 Biblioteki niestandardowe

Program używa biblioteki GTK+ w wersji 3.22.

2 Kompilacja

Do kompilacji programu służy plik Makefile - w linii poleceń należy wpisać polecenie „make", następnie jednokrotnie uruchomić program poleceniem „./Reversi", bez podawania dodatkowych argumentów, celem zainicjowania plików fifo AtoB i BtoA.

3 Uruchomienie programu

W celu rozpoczęcia gry należy uruchomić plik wykonywalny dwukrotnie w katalogu, w którym został utworzony, raz poleceniem „./Reversi A" (dla gracza 1), drugi raz poleceniem „./Reversi B" (dla gracza 2).

4 Moduły programu

Program składa się z 5 modułów: programu głównego main.c, 2 modułów do obsługi komunikacji – linfifo.c i fifo.h oraz 2 modułów zawierających funkcje związane z przebiegiem rozgrywki – reversi.c i reversi.h. W katalogu, w którym znajdują się te pliki, powinny również znaleźć się 4 pliki *.png (red.png, black.png, blank.png i bluesmall.png), zawierające obrazy do wyświetlenia na planszy, oraz stworzone przy pierwszym uruchomieniu programu pliki kolejkowe AtoB i BtoA.

5 Opis modułów

5.1 main.c

Moduł główny; zawiera funkcję main i kilka funkcji operujących na zmiennych zadeklarowanych globalnie. W funkcji `main` znajdują się opisy wszystkich okien i okienek dialogowych - interfejsu użytkownika. Wygląd poszczególnych okienek różni się w zależności od podanego argumentu programu.
Pozostałe funkcje dotyczą zarówno komunikacji między instancjami programu, jak i przebiegu rozgrywki.

- Funkcje `sendText` i `getText` służą odpowiednio do wysyłania i odbierania wiadomości z pliku kolejkowego. Aktualna wiadomość jest przechowywana w programie w zmiennej globalnej message. Dodatkowo funkcja `getText` wykonuje wstępną analizę otrzymanego tekstu i w zależności od otrzymanego komunikatu wywołuje odpowiednie funkcje.
- Funkcje `surrender`, `endMyGame`, `newGameAfterTheEnd`, `canWePleasePlayANewGameNow`, `denyAndReturnToGame`, `agreeAndPlayNew`, `closeAndReturn` służą do tworzenia komunikatów (zgodnych z nazwą funkcji) oraz wywoływania funkcji pomocniczych związanych z odpowiednimi sytuacjami w grze.
- Funkcja `updateNames` aktualizuje nazwy okienek i treści wyświetlanych wiadomości, wykorzystując podane przez obu graczy imiona.
- Funkcje `makeBoard` i `updateBoard` służą odpowiednio do tworzenia nowej planszy i aktualizowania jej stanu. Aktualna sytuacja w grze (tj. informacja dla każdego pola, czy jest ono puste lub czy jest zajęte przez pionek jednego z graczy) jest przechowywana w dwuwymiarowej tablicy game.

- Funkcje `readInfoAndDestroyWindow` oraz `readAndDestroyWindow` służą odczytywaniu informacji wpisanych odpowiednio przez graczy 1 i 2. Jeśli informacje są poprawne, wywoływane są kolejne funkcje, m.in. `makeBoard`, prowadzące do rozpoczęcia rozgrywki, w przeciwnym wypadku gracze są informowani o wprowadzeniu nieprawidłowych danych i mogą je poprawić.
- Funkcja `buttonPressCallback` służy obsługiwaniu kliknięć użytkowników w przestrzeń planszy. W zależności od tego, który z graczy ma w danym momencie prawo ruchu, oraz czy zamierzony przez niego ruch jest poprawny, funkcja dokonuje aktualizacji stanu rozgrywki oraz wywołuje funkcję odpowiedzialną za wysłanie informacji o ruchu drugiemu graczowi, lub, jeśli ruch był niepoprawny, ignoruje kliknięcie gracza.

5.2 fifo.h oraz lin-fifo.c

Funkcje zadeklarowane w pliku nagłówkowym fifo.h i opisane w pliku lin-fifo.c służą obsługiwaniu komunikacji między instancjami programu, m.in. inicjowaniu plików kolejkowych, wysyłaniu i odbieraniu informacji.

5.3 reversi.h oraz reversi.c

Funkcje zadeklarowane w pliku nagłówkowym reversi.h i opisane w pliku reversi.c służą obsłudze rozgrywki.

- Funkcje `can1MakeAMove`, `can2MakeAMove`, `can1MakeAMoveAtAll` `can2MakeAMoveAtAll`, `MakeA1Move`, `MakeA2Move` służą odpowiednio sprawdzaniu, czy gracz 1 lub 2 mogą na przekazanym w argumencie polu ustawić swój pionek (rozważają po kolei wszystkie ewentualne kierunki tworzenia linii pionków przeciwnika), sprawdzaniu czy na planszy istnieje pole, na którym dany gracz może prawidłowo postawić swój pionek oraz aktualizowaniu planszy po wykonaniu ruchu przez danego gracza.
- Funkcja `readInfoAboutAMove` odczytuje z odebranej od drugiego gracza wiadomości informację o wykonanym przez niego ruchu.
- Funkcje `appendDisc` oraz `appendHints` służą aktualizowaniu wyświetlanych na planszy obrazków z pionkami lub podpowiedziami dla graczy.
- Funkcje `endOfTheGame`, `prepareMessage` i `appendMessage` służą liczeniu liczby pionków każdego z graczy, znajdujących się w danym momencie na planszy (również po zakończeniu rozgrywki), przygotowywaniu informacji o podsumowaniu zakończonej gry oraz wpisywaniu tych informacji do odpowiedniego okienka pokazywanego graczom.
- Funkcja `checkNames` porównuje wprowadzone przez graczy imiona, i, jeśli są takie same, dopisuje do nich odpowienio „(player 1)” i „(player 2)”.