 **Технически Университет – София**

**Факултет по компютърни системи и технологии**

**Специалност “Компютърни и Софтуерно Инженерство”, бакалавър**

**КУРСОВА РАБОТА**

**по ПРОГРАМНИ СРЕДИ**

**Студент: Иван Василев Новаков**

**ФАК. No: 121215126 Група: 38**

|  |
| --- |
|  |
| Направете игра наподобяваща по правила играта Breakout/Bricks |

Подпис на студента: Подпис на преподавателя:

**……………………** **……………………**

Играта е реализирана чрез WPF.

**Глобални променливи:**

private List<Rectangle> bricks = new List<Rectangle>();

Списък от всички тухли.

private int curretLvl = 0;

Текущо ниво.

private int ball\_speed;

Скорост на топчето.

private bool isClockWise = true;

Променлива, която се използва при промяна на посоката на движение на топчето след сблъсък. Проверява дали промяната ще е по часовниковата стрелка.

private int currentDirection = 0;

Текуща посока на движение. В момента посоките са 4. По 4-те диагонала.

private string[] levelsInfo;

Съдържа данните за нивата. Данните се прочитат от текстов файл с име GameLevels. Всяко ниво е представено като таблица 10x10. Клетките/тухлите са разделени със ‘,‘. Всяка клетка може да има стойност интервал, 1, 2, или 3.

private string[] brickInfo;

Съдържа данните за тухлите. В началото на всяко ниво се зарежда от levelsInfo. Променя се при всяка ударена тухла.

private DispatcherTimer movingTimer = new DispatcherTimer();

Декларация на таймера, който отговаря за движението на топчето.

Rectangle lastCollapsed = default(Rectangle);

Съдържа последната докосната тухла.

**Функции:**

private void Window\_MouseMove(object sender, MouseEventArgs e)

Функция, която следи движението на мишката. Взимат се координатите на курсора и се пращат към SetPaddlePosition().

private void ClearCanvas()

Скрива всички елементи от списъка с тухли, и изтрива цялото съдържание на списъка.

public void SetInitialState()

Връща играта в начално състояние (скоростта на топчето, координатите му, посоката му на движение).

private void DispatcherTimer\_Tick(object sender, EventArgs e)

Таймер, който на всеки 10ms проверява посоката на двиение на топчето, обноявва неговата позиция и проверява за контакт с друг обект.

private void StartGame(object sender, RoutedEventArgs e)

Функция, която се извиква при натискане на бутона “Start Game”. Стартира таймера, който отговаря за движението на топчето.

public void MoveGameBall()

Функция, която променя позицията на топчето спрямо предишната му позиция, неговата скорост и посоката му на движение.

public void UpdateDirection()

Обновява посоката на движение на топчето, спрямо това дали се е ударило в някоя от стените.

private bool CheckBottomBreakCollapse()

Проверява дали топчето се е ударило в подложката или в долната стена(край на играта).

private List<Coordinates> GetCircularPoints()

Изчислява координатите(360 точки) по периметъра на топчето и ги записва в pointLists списъка.

public void ChangeBallDirection(Rectangle crashedBrick, Coordinates nearCoordinate)

Тази функция се вика при сблъсък на топчето с някоя тухла. Получава като параметри тухлата, която е ударен и координатите на точката, в която е удара. Спрямо тях се определя посоката на движение на топчето след сблъсъка.

private void BrickGenerator(int currentLvl)

Според даденото ниво(currentLvl) и съдържанието на levelsInfo, генерира тухлите за нивото. Ако няма данни за въведеното ниво, спира таймера и изкарва съобщение “You Won!”.

private List<Coordinates> GetBorderPoints()

Създава списък от координати с граничните точки на топчето.

private bool CheckBreakCollapse()

Проверява за сблъсък между топчето и някоя тухла. При сблъсък премахва тухлата или променя цвета ѝ (в зависимост от типа ѝ), променя посоката на движение на топчето и изкарва съобщение за успешно преминаване на нивото, ако няма повече тухли.

public void SetPaddlePosition(double positionX)

Функция, която задава нова позиция на подложката по зададена координата по X. Функцията се вика при движение на мишката.