

**ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ**

**Курсова работа**

**по**

**Програмни среди**

**Pong game**

Изготвена от:

* Ивайло Христофоров Христов , група 38, фак. № 121215077

**Задание**

Реализация на играта Pong чрез WPF и C#.

**Реализация**

Играта Pong представлява 2D игра, наподобяваща тенис на маса.

1. XAML реализация:

Приложението се състои от един MainWindow.xaml.

В него се намира един Canvas с размери 800х500 пиксела, в който ще се изрисуват играчите и топчето.

2. Code-Behind реализация:

2.1 Клас MainWindow

Това е класът, който стои зад XAML-a. В този клас се прихващат евентите от бутоните на клавиатурата. При инициализирането на класа се стартира таймер-а, настроен на 25 милисекунди. Когато таймерът изтече, програмата обновява позицията на играчите и топчето, и обновява екрана.

Класът има 2 частни полета от тип PongGame и DispatcherTimer.

Методи:

*public MainWindow()* - конструктор

void timer\_Timeout(object sender, EventArgs e) – метод,който се извиква при изтичане на таймера.

*private void Window\_KeyDown\_1(object sender, KeyEventArgs e)* – евент при натискане на клавиш. Информацията за клавишите се изпраща към PongGame обекта.

*private void Window\_KeyUp\_1(object sender, KeyEventArgs e)* – евент при пускането на натиснат клавиш. Информацията за клавишите се изпраща към PongGame обекта.

*private void Window\_Loaded(object sender, RoutedEventArgs e)* – евент при зареждане на приложението. В този евент се стартира таймера и се извиква updateDisplay() метода.

*private void updateDisplay()* – метод за изчертаване на играчите и топчето.

2.2. PongGame

Този клас отговаря за играчите, топчето и резултата на играта.

Класът има 2 полета от тип Player за всеки от двамата играчи. Едно поле от тип Ball за игралното топче. Две полета от тип int, където ще се съхранява резултата на играчите. Едно поле от тип int, което ще се обновява, когато играта приключи.

Методи:

*public PongGame(double windowHeight, double windowWidth)* – констуктор

*public void update\_position()* – този метод се извиква от MainWindow класа. В него се извиква метод за обновяване не позицията на топчето. След това се проверява дали играта е приключила. Ако играта продължава, се извиква метод за обновяване на позицията на двамата играчи.

*public void resetBall()* – в този метод се извиква метод за рестартиране на положението на топчето.

*public void update\_key(KeyEventArgs e, bool down)* – в този метод се изпраща информацията към всеки играч за обновяване на състоянието на клавишите на клавиатурата.

*public List<Shape> getAllShapes()* – този метод добавя в списък всички фигури, които визуализират играта.

2.3 Клас Player

Класът интерпретира играча и неговата позиция.

Той пази в себе си мембър от тип Rectangle, който ще служи за визуализиране на плочката на играча. Клавишите, които движат играча. Флагове, които пазят статус на клавишите. Позицията на играча спрямо х и у. Размерът на игралното поле и големината на стъпката, с която играчът се движи.

Mетоди:

*public Player(double windowHeight, Key upKey, Key downKey, int x)* - контруктор

*public void reset\_key()* – служи за зачистването на полетата на класа, които пазят състоянитео на клавишите, които управляват играча.

*public void update\_pos()* – обновява положението на играча спрямо състоянието на клавишите, които управляват играча.

*public void update\_key(Key key, bool down)* – обновява състоянието на клавишите, които управляват играча.

**Начин на работа**

Играч 1 е този в син цвят. Той се управлява с клавишите W(нагоре) и S(надолу).

Играч 2 е този в червен цвят. Той се управлява с клавишите ↑(нагоре) и ↓ (надолу).

При изтичане на таймер-а:

Обновява се положението на топчето. Ако е отбелязан гол, се обновява резултата и топчето се поставя в начална позиция.

Ако няма гол, се обновява позицията на топчето и позицията на играчите в зависимост от натиснатите клавиши.

След това се изчертават играчите и топчето с новите си позиции.