Министерство Образования Республики Беларусь Белорусский  
Государственный Университет Информатики и Радиоэлектроники  
Кафедра Информатики

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
К КУРСОВОМУ ПРОЕКТУ  
Дисциплина: Программирование  
Тема: Battleships

Принял:  
Козуб Виктор Николаевич  
Выполнил: студент гр.653501  
Лескевич Антон Казимирович

Минск 2017

Содержание  
1. Введение…… ……………………………………………........3  
2. Сравнение с “другим”морским боем…..…..4  
2.1. Функциональность… …………………...………….......4  
2.2. Интерфейс ...………………….………...…………...…..4

2.2. Производительность ...………………….………...…………...…..4  
2.4. История ...……………………..…………...……..….…..5  
3. Скриншоты и описание ...……………………..………......….6  
4. Техническая информация ....……………...………........…….11  
4.1. Средства разработки ...……………………….......…….11  
4.2. Структура программы ...……………………...….……..11  
5. Описание работы программного кода ………………………12  
6. Тестирование …………….…………………......…………...20  
7. Выводы …...………………………………..………………...21  
8. Содержание ..………………………………………………...22

1.Введение

Конечной целью этого проекта является создание игры “морской бой” для  
платформы Windows. Игры, которая должна понравиться большинству пользователей, которая с может их затянуть и помочь незаметно скоротать время, при этом получить уйму удовольствия. Игра также должна удовлетворять пользователей своей функциональностью (возможность расстановки кораблей как вручную, так и автоматически, таблица рекордсменов по победным играм, а также музыкой, которая делала процесс игры более приятным).

В ходе разработки этого проекта на практике должны закрепиться основы разработки приложений в WPF (Windows Presentation Foundation). Возможно будут выявлены основные проблемы, характерные при работе с подобными приложениями, тем самым нам понадобятся определённые знания, необходимые для их решения. В результате процесс разработки данного приложения упростит для нас более глубокое изучение данной темы в будущем.

2. Сравнение с “другим” морским боем

2.1. Функциональность  
Данный проект значительно уступает продукту морской бой в функциональности. Так как данная игра была разработана в очень короткие сроки, в нём содержится только базовый функционал по расстановке, стрельбе и игре в целом.

2.2. Интерфейс  
Обе программы обладают достаточно дружелюбным интерфейсом.  
Однако стоит отметить, что в связи с повышенной функциональностью аналог Морской бой 2 боле приятный для игры, так как в нём присутствуют различные анимации и дополнительные бонусы при игре. Однако стоит отметить то, что интерфейс данного игры очень  
прост, что позволяет пользователю без затруднений разобраться с  
функциональностью программы. Что касается языков интерфейса, для  
Batleships разработан только английский интерфейс, в то время, как в аналоге имеются и другие языки.

2.3.Производительность

Игра Морской бой 2 потребляет в среднем около 100мб оперативной памяти, в то время как игра, разработанная мной 30мб. Однако подобные отличия не так важны в современном мире, так как они не будут влиять на производительность системы значительным образом.

2.4.История

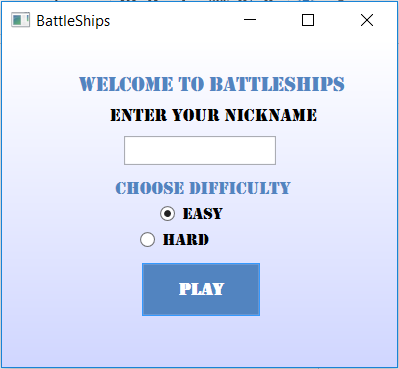
Игра Морской бой 2 от компании Byril Games вышла на экраны игровой индустрии в 2014 году, в то время как первая версия существовала на рынке уже достаточно много времени.

Данная игра превысила отметку скачиваний в 10 млн, что говорит о её функциональности и приятном интерфейсе.

Также достаточно популярны игры от этой компании такие как, Алхимия на бумаге, Точки онлайн и другие.

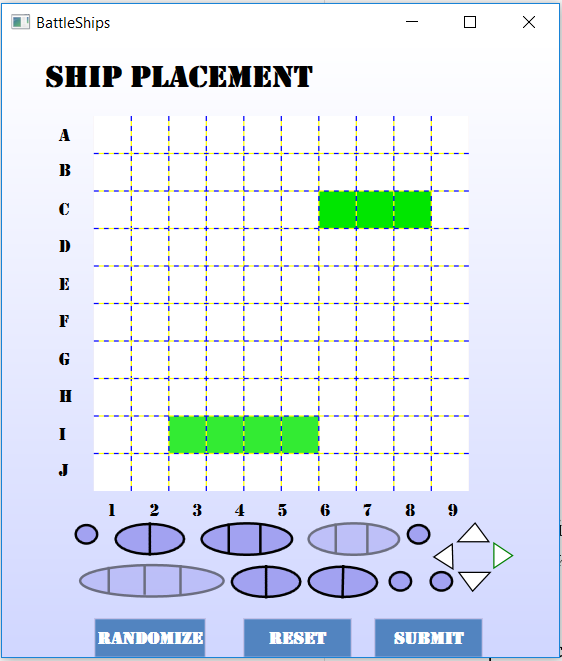
Что же касается игры Battleship, то на его разработку было потрачено порядка 3 недель, что говорит о том, что данная игра может быть расширена за счёт небольшого количества составляющих.

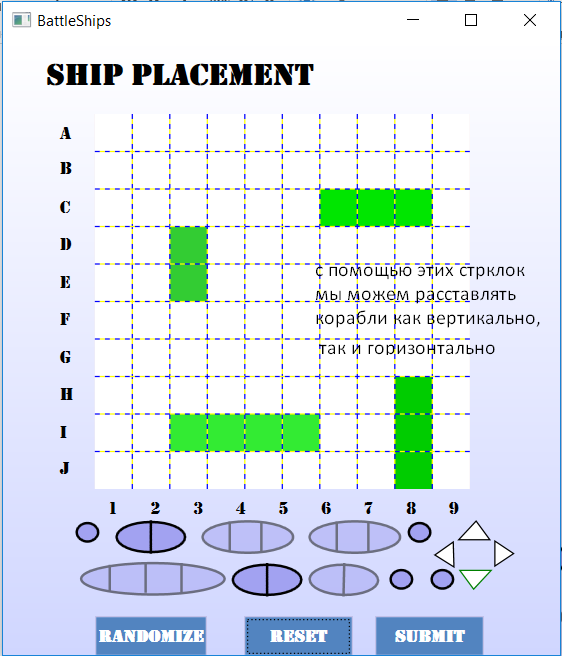
3. Скриншоты и описание



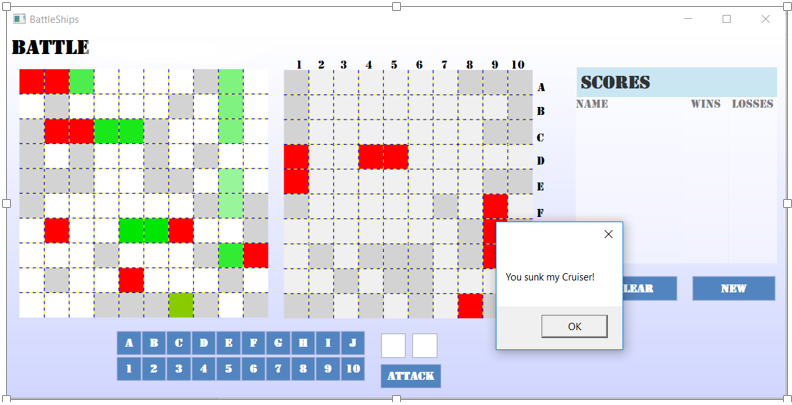
Интерфейс программы простой и довольно понятный. При запуске приложения мы должны ввести своё имя и выбрать уровень сложности.

После этого мы сможем начать расстановку наших кораблей.



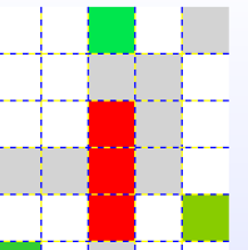


Также мы можем воспользоваться кнопкой RANDOMIZE, чтобы программа расставила корабли вместо нас.

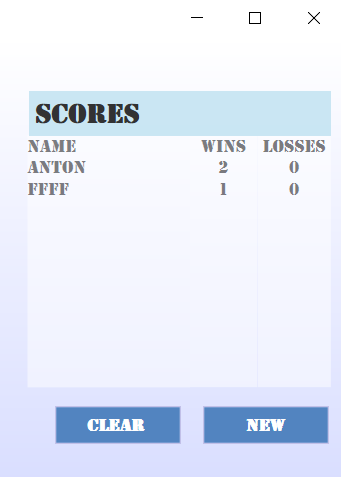


Дальше собственно и начинается игра. Мы можем стрелять по полю противника как нажатием на определённый квадратик, так и использовать функцию ATTACK, где нам надо вводить координаты поля, куда мы собираемся нанести свой удар.

Отличие лёгкого режима от сложного заключается в том, что в лёгком противник всегда стреляет рандомно, а в сложном найдя часть корабля, продолжает его уничтожение.



При уничтожении вражеского корабля, или когда противник уничтожит ваш, будет выслано уведомление у потере своего или уничтожении вражеского корабля.



Эта часть экрана приложения выполняет роль таблицы игроков, которая показывает лучших игроков и долю их побед и поражений.

Также этот список можно очистить.

4. Техническая информация

4.1. Средства разработки  
Для разработки приложения использовалась система для построения  
клиентских приложений Windows с использованием WPF (Windows Presentation Foundation), графическая подсистема в составе .NET Framework, использующая зык XAML.

WPF может быть использована в WINDOWS VISTA, 7, 8, 10, с помощью которой можно создавать широкий спектр приложений, как автономных, так и запускаемых в браузере.

Разработка велась в среде Visual Studio 2015, так как данная среда является достаточно новым и полноценным средством работы с платформой .NET и поставляется бесплатно в редакции Community.

4.2. Структура программы  
MainWindow.xaml – тут содержится код главного окна программы.  
MainWindow.xaml.cs – файл с кодом на C#, где содержится код программы.  
Start.xaml – тут содержится код окна, которое мы видим при запуске программы.  
Start.xaml.cs – файл с кодом на C#, для начала игры.

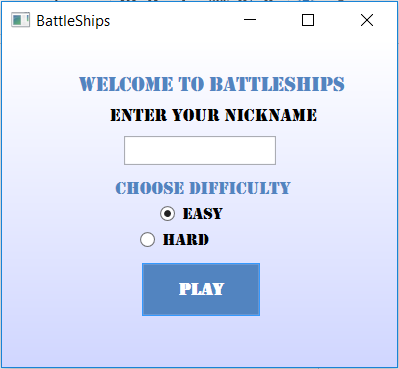
VSComp.xaml – тут содержится код окна для непосредственной игры.

VSComp.xaml.cs – файл с кодом на C#, где содержатся все действия для игры против компьютера.

Ships.xaml – тут содержится код окна программы, когда производится расстановка кораблей.

Ships.xaml.cs – файл с кодом на C#, где описывается расстановка кораблей.

5. Описание работы программного кода  
В начале требуется запустить Battleships.exe.



public partial class Setup : UserControl

{

public event EventHandler play;

public string name;

public Difficulty difficulty = Difficulty.Simple;

public Setup()

{

InitializeComponent();

}

private void buttonStart\_Click(object sender, RoutedEventArgs e)

{

name = txtboxName.Text;

if (name == "")

{

MessageBox.Show("You must enter a name", "Error", MessageBoxButton.OK,MessageBoxImage.Error);

}

else

{

play(this,e);

}

}  
Требуется ввести имя, а то игра не запустится.

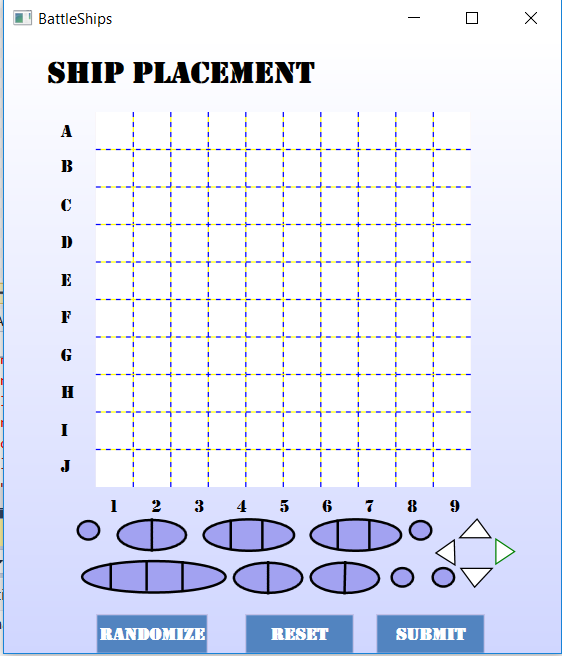
private void rbtnSimple\_Click(object sender, RoutedEventArgs e)

{

difficulty = Difficulty.Simple;

}

Выбираем сложность, аналогично и для того, чтобы выбрать “HARD”.



Сначала создаётся поле 10\*10. Для каждого корабля создаётся отдельный цвет, а аткже цвет для моря.

После этого, выбираем корабль, который хотим разместить

private void ship\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)

{

Path shipPath = (Path)sender;

if (!shipPath.IsEnabled)

{

return;

}

if (lastShip != null)

{

lastShip.Stroke = unselected;

}

Теперь выбираем направление (по горизонтали или вертикали).

private void orientationMouseDown(object sender, MouseButtonEventArgs e)

{

Polygon arrow = (Polygon)sender;

lastArrow.Stroke = unselected;

lastArrow = arrow;

arrow.Stroke = selected;

if (arrow.Name.Equals("rightPoly") || arrow.Name.Equals("leftPoly"))

{

orientation = Orientation.HORIZONTAL;

}

else

{

orientation = Orientation.VERTICAL;

}

}

После этого происходит проверка достаточно ли места и сам процесс размещения корабля.

Также может проводиться автоматическая установка кораблей.

Если эти все требования выполнены, то мы можем начать непосредственно играть.

private void btnSubmit\_Click(object sender, RoutedEventArgs e)

{

if (numShipsPlaced != 10)

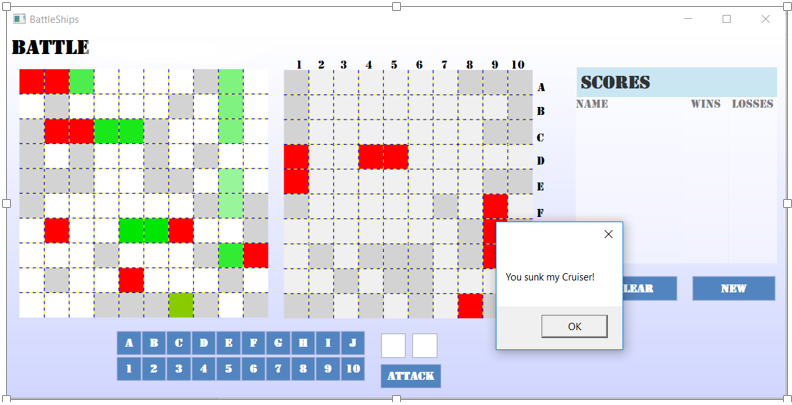
{

return;

}

play(this,e);

}



Далее мы начинаем обстрел поля противника, в надежде потопить его корабли.

private void gridMouseDown(object sender, MouseButtonEventArgs e)

{

Grid square = (Grid)sender;

if (turnCount % 2 != 0)

{

return;

}

switch (square.Tag.ToString())

{

case "water":

square.Tag = "miss";

square.Background = new SolidColorBrush(Colors.LightGray);

turnCount++;

compTurn();

return;

case "miss":

case "hit":

Console.WriteLine("User hit a miss/hit");

return;

case "small":

cSmallCount--;

break;

case "small\_Copy":

cSmall\_CopyCount--;

break;

case "small\_Copy1":

cSmall\_Copy1Count--;

break;

case "small\_Copy2":

cSmall\_Copy2Count--;

break;

case "destroyer":

cDestroyerCount--;

break;

case "destroyer\_Copy":

cDestroyer\_CopyCount--;

break;

case "destroyer\_Copy1":

cDestroyer\_Copy1Count--;

break;

case "cruiser":

cCruiserCount--;

break;

case "submarine":

cSubmarineCount--;

break;

case "carrier":

cCarrierCount--;

break;

}

Если попадаешь, то клетка становится красной. Также после кадого действия проверяется победил игрок или нет.

square.Tag = "hit";

square.Background = new SolidColorBrush(Colors.Red);

turnCount++;

checkPlayerWin();

compTurn();

}

private void checkPlayerWin()

{

if (cCarrierCount == 0)

{

cCarrierCount = -1;

MessageBox.Show("You sunk my Aircraft Carrier!");

}

if (cCruiserCount == 0)

{

cCruiserCount = -1;

MessageBox.Show("You sunk my Cruiser!");

}

if (cDestroyerCount == 0)

{

cDestroyerCount = -1;

MessageBox.Show("You sunk my Destroyer!");

}

if (cSmall\_CopyCount == 0)

{

cSmall\_CopyCount = -1;

MessageBox.Show("You sunk my small!");

}

if (cSmall\_Copy1Count == 0)

{

cSmall\_Copy1Count = -1;

MessageBox.Show("You sunk my small!");

}

if (cSmall\_Copy2Count == 0)

{

cSmall\_Copy2Count = -1;

MessageBox.Show("You sunk my small!");

}

if (cSmallCount == 0)

{

cSmallCount = -1;

MessageBox.Show("You sunk my small!");

}

if (cDestroyer\_CopyCount == 0)

{

cDestroyer\_CopyCount = -1;

MessageBox.Show("You sunk my Destroyer!");

}

if (cDestroyer\_Copy1Count == 0)

{

cDestroyer\_Copy1Count = -1;

MessageBox.Show("You sunk my Destroyer!");

}

if (cSubmarineCount == 0)

{

cSubmarineCount = -1;

MessageBox.Show("You sunk my Submarine!");

}

if (cCarrierCount == -1 && cDestroyer\_CopyCount == -1 && cDestroyer\_Copy1Count == -1 && cSubmarineCount == -1 &&

cCruiserCount == -1 && cDestroyerCount == -1 && cSmallCount == -1 && cSmall\_CopyCount == -1 && cSmall\_Copy1Count == -1 && cSmall\_Copy2Count == -1)

{

MessageBox.Show("You win!");

disableGrids();

displayHighScores(saveHighScores(true));

}

Аналогично происходит и с игрой компьютера.

В правой стороне показан список лучших игроков.

private void displayHighScores(List<string> players)

{

string[] player;

string names = "Name" + Environment.NewLine;

string wins = "Wins" + Environment.NewLine;

string losses = "Losses" + Environment.NewLine;

for (int i = 0; i < players.Count; i++)

{

player = players[i].Split(' ');

names += player[0] + Environment.NewLine;

wins += player[1] + Environment.NewLine;

losses += player[2] + Environment.NewLine;

}

txtBlockNames.Text = names;

txtBlockWins.Text = wins;

txtBlockLosses.Text = losses;

}

Когда игрок начинает игру, происходит бинарный поиск его в списке, если его там не оказывается, то создаём

if (!File.Exists(filename))

{

FileStream stream = File.Create(filename);

stream.Close();

}

По окончанию игры происходит изменения в списке с целью записи новых результатов.

Также у нас есть возможность начать новую игру

private void btnStartOver\_Click(object sender, RoutedEventArgs e)

{

replay(this, e);

}

То есть мы можем играть не закрывая приложение сколько душе угодно.

|  |
| --- |
|  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

6. Тестирование  
Работа программы была проверена на операционной системе Windows 8, 10. Также работа программы была проверена на всех возможных  
разрешениях экрана, что также подтвердило ее стабильность. Во время  
тестирования ошибок в ходе программы не было обнаружено.

7. Выводы  
В ходе работы над проектом были изучены и закреплены на практике  
основные принципы работы в WPF. Был выявлен ряд проблем, часть из которых были решены во время работы над приложением. Среди этих  
проблем: действия искусственного интеллекта, подтверждения уничтожения корабля противника.

Были определены проблемы, требующие дальнейшего изучения. Например, сохранение чистоты и, соответственно обслуживаемости кода. В данном проекте это не совсем получилось, из-за отсутствия опыта работы в WPF в прошлом, недостаточного времени, отведенного на проектирование программы и неполное знание средств выбранной системы. Также стала очевидной необходимость придерживаться заранее определенного стиля написания кода для облегчения его читаемости.  
Несмотря на некоторые недостатки в программном коде, поставленные цели были достигнуты. Программа отлично работает в независимости о программного и аппаратного обеспечения. Из всего выше перечисленного считаю проект очень полезным, а также успешно завершённым.

Список литературы  
1. <https://ru.wikipedia.org/wiki/Windows_Presentation_Foundation>  
2. <https://metanit.com/sharp/wpf/>  
3. <https://msdn.microsoft.com/ru-ru/library/ms752059(v=vs.110).aspx>  
4. <https://ru.wikipedia.org/wiki/XAML>  
5. <https://metanit.com/sharp/wpf/2.php>