ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук

Департамент программной инженерии

|  |  |
| --- | --- |
| СОГЛАСОВАНО  Доцент департамента  программной инженерии  факультета компьютерных наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_О.В.Максименкова  « \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. | УТВЕРЖДАЮ  Академический руководитель  образовательной программы  «Программная инженерия», профессор,  канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  « \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. |

|  |  |
| --- | --- |
| Инв. № подл. |  |
| Подп. и дата |  |
| Взам. Инв. № |  |
| Инв. № дубл. |  |
| Подп. и дата |  |

**Трёхмерная компьютерная игра Маджонг**

**Пояснительная записка**

**ЛИСТ УТВЕРЖДЕНИЯ**

**RU.17701729.04.01-01 81 01-1-ЛУ**

Исполнитель

студент группы БПИ181

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/А.И. Уварова/

« \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

**Москва 2019**

УТВЕРЖДЕН

RU.17701729.04.01-01 81 01-1 ЛУ

**Трёхмерная компьютерная игра Маджонг**

**Пояснительная записка**

|  |  |
| --- | --- |
| Инв. № подл. |  |
| Подп. и дата |  |
| Взам. Инв. № |  |
| Инв. № дубл. |  |
| Подп. и дата |  |

**RU.17701729.04.01-01 81 01-1**

**Листов 51**

**Москва 2019**

**Содержание**

[**1.ВВЕДЕНИЕ** 3](#_Toc8400542)

[**1.1 Наименование программы** 3](#_Toc8400543)

[**1.2 Основания для разработки** 3](#_Toc8400544)

[**2.Назначение и область применения** 4](#_Toc8400545)

[**2.1 Назначение программы** 4](#_Toc8400546)

[**2.2 Краткая характеристика области применения** 4](#_Toc8400547)

[**3. Технические характеристики** 5](#_Toc8400548)

[**3.1 Постановка задачи на разработку программы** 5](#_Toc8400549)

[**3.1.1 Описание правил игры [3]** 5](#_Toc8400550)

[**3.2 Описание алгоритма и функционирования программы** 10](#_Toc8400551)

[**3.2.1 Описание функционирования программы** 10](#_Toc8400552)

[**3.2.2 Описания алгоритма проверки закрытых костей при объявлении Маджонга** 11](#_Toc8400553)

[**3.3 Описание входных и выходных данных** 14](#_Toc8400554)

[**3.4. Описание и обоснование выбора состава технических средств** 15](#_Toc8400555)

[**3.5** **Описание и обоснование выбора состава программных средств** 15](#_Toc8400556)

[**4.ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ** 16](#_Toc8400557)

[**5. ПРИЛОЖЕНИЯ** 17](#_Toc8400558)

[**ПРИЛОЖЕНИЕ 1** 17](#_Toc8400559)

[**ПРИЛОЖЕНИЕ 2** 18](#_Toc8400560)

[**ПРИЛОЖЕНИЕ 3** 20](#_Toc8400561)

[**ПРИЛОЖЕНИЕ 4.** 22](#_Toc8400562)

# **1.ВВЕДЕНИЕ**

## **1.1 Наименование программы**

Наименование программы – «Трёхмерная компьютерная игра Маджонг».

Наименование на английском языке – «3D PC-game Mahjong».

## **1.2 Основания для разработки**

**Документ, на основании которого ведется разработка:**

Программа выполнена в рамках задания на курсовую работу в соответствии с учебным планом подготовки бакалавров (НИУ ВШЭ, факультет компьютерных наук) по направлению «Программная инженерия». Приказ НИУ ВШЭ № 2.3-02/1012-0 2 от 10.12.18.

**Организация, утвердившая документ:**

Документ утвержден НИУ ВШЭ, факультетом компьютерных наук, департаментом программной инженерии.

# **2.Назначение и область применения**

## **2.1 Назначение программы**

Программа разработана для развлекательных целей, а также для развития логики и стратегического мышления.

## **2.2 Краткая характеристика области применения**

Программа предназначена для игры в настольную китайскую игру «Маджонг» on-line. Данная программа подойдет любителям Маджонга, а также людям, увлекающимся логическими или азартными играми. Игра также отлично подойдет для компании людей, у которых нет времени собраться вместе или отсутствует стандартная настольная версия игры.

# **3. Технические характеристики**

## **3.1 Постановка задачи на разработку программы**

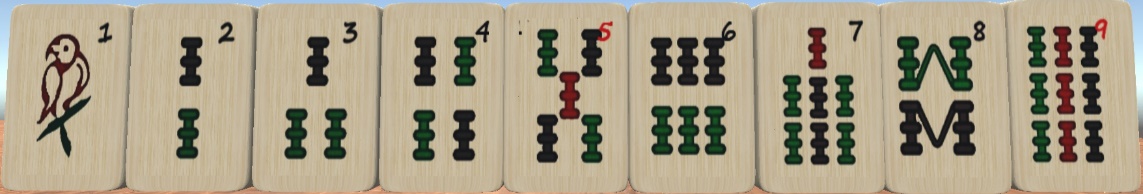
«Маджонг» - китайская азартная настольная игра. У нее мало компьютерных аналогов в интернете, а у тех, что существуют, либо плохое качество, либо отсутствует возможность онлайн-игры, либо за игру необходимо платить. Поэтому была поставлена задача разработать программу для игры в «Маджонг», которая бы соответствовала заявленным в техническом задании требованиям.

### **3.1.1 Описание правил игры [3]**

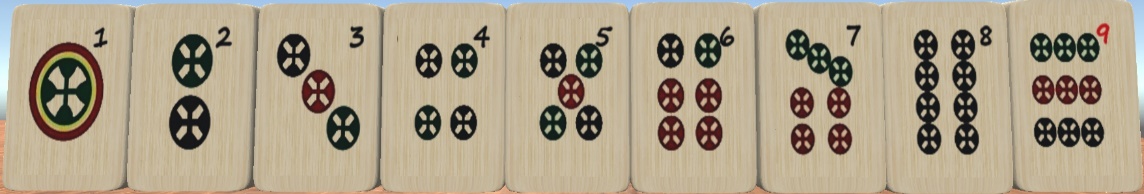
**Описание игровых костей**

Игра ведется набором из 144 костей. В полный набор входит по 4 одинаковых кости каждого из 34 видов + 8 дополнительный костей.

34 вида костей складываются из: трех мастей -Бамбук, Доты и Символы (см. рис.1, 2 и 3). В каждой из трех мастей кости идут от единицы до девятки.



*Рисунок 1. Бамбук*

**

*Рисунок 2. Доты*

**

*Рисунок 3. Символы*

Кроме этого, в игре есть четыре ветра - Восточный, Южный, Западный и Северный (см. рис. 4), а также три Дракона – Красный, Зеленый и Белый (см. рис. 5).



*Рисунок 4. Ветра*

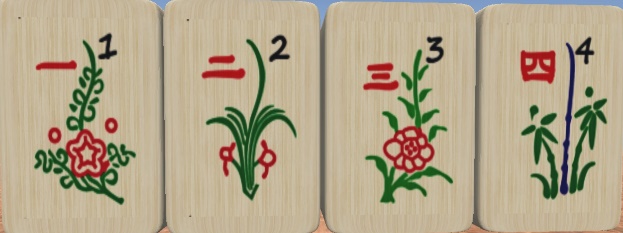
**

*Рисунок 5. Драконы*

Последние 8 костей – Сезоны и Цветы (см. рис. 6,7).



*Рисунок 6. Сезоны*

**

*Рисунок 7. Цветы*

**Церемония начала игра**

Для игры в Маджонг необходимо 4 человека. Каждому игроку случайным образом (в оригинале с помощью игральных кубиков) присваивается одна из 4-х сторон света: Восток (лидер), Юг, Запад, Север.

В начале игры кости выкладываются на стол лицевой стороной вниз, перемешиваются, затем каждый игрок выкладывает перед собой 36 костей лицевой стороной вниз в ряд из 18 костей в длину и 2 костей в высоту. Затем 4 ряда сдвигаются в центр так, чтобы образовался квадрат, символизирующий китайскую городскую стену (рис. 8).

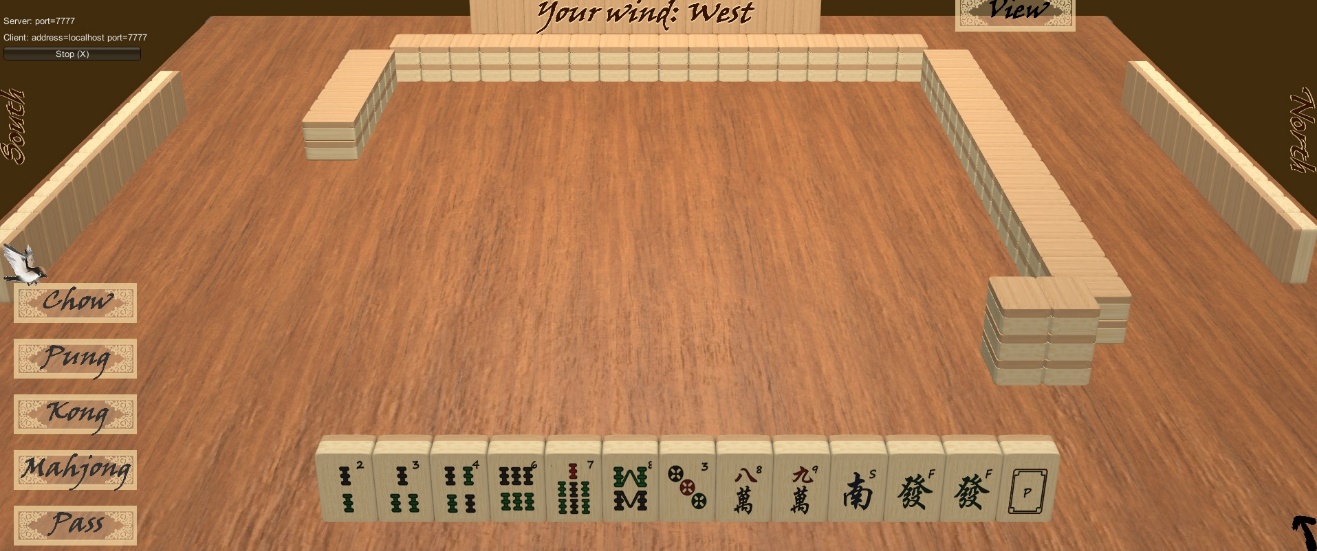


*Рисунок 8. Стена*

**Разбор Стены**

Затем случайным образом (в оригинале с помощью игральных кубиков) определяется одна из четырех сторон стены и место, где Стена вскрывается. Две первые вынутые из Стены кости, называемые свободными костями, кладутся третьим рядом справа от места разбора лицевой стороной вниз. В дальнейшем свободные кости будут указывать конец Стены.

Далее лидер (Восточный ветер) берет первые 4 кости (2 пары) с левой части открытой Стены. За ним по очереди берут по 4 кости Юг, Запад и Север. Таким образом игроки разбирают стену до тех пор, пока у каждого не будет 12 костей. На четвертом заходе таким же образом каждый игрок берет еще по одной кости. Таким образом, в начале игры у каждого игрока по 13 фишек. В случае, если у кого-то из игроков в начале игры ( при взятии фишки со стены в дальнейшем тот же алгоритм) в наборе есть Цветок или Сезон, игроки в той же очереди выкладывают их на стол перед собой и берут недостающие фишки из свободных костей (когда свободные кости кончаются, кладется следующая пара с конца стены). После этого стена выглядит примерно так, как на рис. 9



*Рисунок 9. Стена после раздачи*

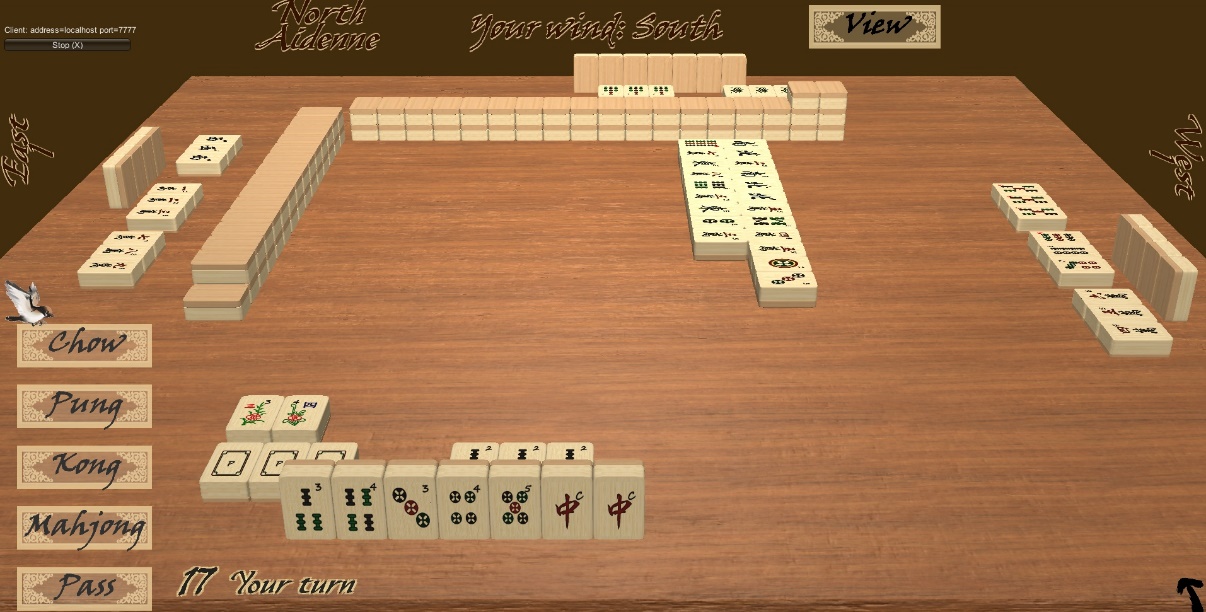
**Игра**

Восток начинает игру, беря 14-ю фишку со стены и выкладывая лишнюю кость на кон в центр стола. За ним такой же ход делают по очереди Юг, Запад и Север. (см. рис. 10) Процесс игры заключается в том, что игроки поочередно заменяют свои кости на более подходящие (к 13 костям берется 14-я и делается снос ненужной кости). Игроки по очереди делают ход либо пока один из них не соберет выигрышную комбинацию (Маджонг), либо пока в стене не останется 14 костей без учета Свободных (в последнем случае объявляется ничья).

Цель игры состоит в том, чтобы собрать выигрышную комбинацию из 4-х групп по 3 или 4 кости и пары одинаковых костей. В группе может быть 3 или 4 одинаковых кости или последовательность из трех костей одной масти.

Число игр не ограничено. Обычно играется один гейм, состоящий из 4-х раундов (Восточный, Южный, Западный и Северный). Раунд заканчивается, когда каждый из игроков по очереди побывает Востоком; соответственно, гейм состоит не менее чем из 16 игр. После каждого раунда осуществляется подсчет очков на основе собранных игроками комбинаций (см. Приложение 1, 2). В конце оговоренного количества игр побеждает игрок, набравший наибольшее количество очков.

Каждому раунду соответствует ветер, который является для всех игроков преимущественным. Кроме того, в каждой игре у каждого игрока есть свой ветер, определяемый положением игрока относительно лидера. Если лидер собирает выигрышную комбинацию, то переход ветров не осуществляется. Иначе ветер у каждого игрока меняется по схеме Восток->Север->Запад->Юг->Восток.



*Рисунок 10. Ход игры*

**Комбинации**

**Чоу**

После того, как любой игрок делает ход, выкладывая кость на кон, сидящий справа от него игрок, чья очередь ходить, имеет право забрать эту кость в том случае, если она образует последовательность с уже имеющейся у него парой костей. В таком случае он собирает «Чоу», выкладывает эти три кости перед собой и делает свой ход.

Игрок обязан уступить эту кость, если есть игрок, который хочет взять ее на Панг/Конг/Маджонг.

**Панг**

Если у игрока есть пара одинаковых костей, а другой игрок сносит такую же третью, то он имеет право собрать «Панг», даже если не его очередь делать ход. Игрок выкладывает три кости перед собой лицевой стороной вверх и делает ход. В таком случае ход переходит к игроку справа, то есть часть игроков может пропустить ход.

**Конг**

Если у игрока есть 3 одинаковых кости, а другой игрок сносит такую же четвертый, то этот игрок имеет право собрать открытый Конг. В этом случае он выкладывает все 4 кости перед собой лицевой стороной вверх и берет одну из свободных костей, а затем делает ход. Игрок не может взять кость другого игрока, если три таких же кости уже выложены на стол в комбинации «Панг». Если игрок имеет 3 одинаковых открытых кости и берет 4-ю со Стены, то он может доложить ее к трем имеющимся, образуя Конг (затем он должен взять свободную кость).

Если игрок собрал все 4 кости со Стены, то он может выложить их перед собой и взять свободную кость, чтобы сделать ход. В этом случае собирается закрытый Конг. Две кости выкладываются лицевой стороной вверх, а две-вниз, чтобы показать, что Конг закрытый. (имеет значение при подсчете очков, см. Приложение 2)

Игроки, собирающиеся взять кость на Конг или Панг обязаны уступить ее игроку, который хочет взять эту кость для Маджонга.

**Маджонг**

Когда игрок завершает комбинацию из 4-х групп по 3 (или 4) кости и пары одинаковых костей, взяв последнюю кость с кона или со Стены, он объявляет «Маджонг». Все игроки открывают свои кости, подсчитываются очки.

Если одна и та же кость подходит для Маджонга нескольким игрокам, выигравшим считается тот, кто сидит ближе всех по очереди к сделавшему снос игроку.

## **3.2 Описание алгоритма и функционирования программы**

### **3.2.1 Описание функционирования программы**

1. При запуске программы перед пользователем открывается главное меню. В меню игрок может изучить правила игры, настроить звуковое сопровождение, выбрать язык. При нажатии на кнопку «Играть» или «Play» игрок переходит в игровое лобби.
2. В лобби игрок может либо создать новую игру, либо присоединиться к уже существующей, выбрав ее из списка.
3. При начале игры загружается игровая сцена. Случайным образом определяется ветер игрока. С помощью скрипта GameMaster хранится словарь всех подключенных игроков, их количество, отслеживается состояние сцены. Как только все 4 игрока подключаются к игре, начинается игровой процесс.
4. Для начала игры необходимо, чтобы все игроки подтвердили готовность, нажав кнопку в специальном всплывающем окне. Если кто-то из игроков не готов, он автоматически удаляется из матча, затем оставшиеся игроки вновь ждут недостающего человека (если это был хост – все игроки возвращаются в лобби)
5. После подтверждения готовности с помощью скрипта BuildWall строится стена. Состояние игры контролируется в GameMaster, который отсылает команды GameManager – основному скрипту, отвечающему за игру. Скрипт Wall контролирует состояние стены, с помощью него раздаются кости игрокам в начале игры. В нем же хранится список всех костей Стены.
6. У каждого игрока 3 списка костей: закрытые кости, цветы, открытые комбинации. После раздачи костей у всех игроков проверяется по очереди наличие цветов или сезонов, если они есть - выкладываются, берутся недостающие кости из свободных. Процесс продолжается до тех пор, пока у всех игроков не останется цветов и сезонов.
7. Начинается игровой процесс. Восток берет кость со стены, она же автоматически «выбирается». Запускается обратный отсчет времени: у игрока есть 30 секунд на ход. Для того, чтобы вынести кость на кон, игрок должен сначала выбрать ее (левая кнопка мыши), затем повторно нажать левую кнопку мыши. Если у игрока заканчивается время на ход, автоматически сносится кость, которая выбрана в данный момент времени.
8. После сноса у всех остальных игроков запускается обратный отсчет: 20 секунд. Если игрок хочет собрать комбинацию, он должен нажать соответствующую кнопку в левом нижнем углу экрана. Также игрок может пропустить ход, нажав кнопку «Пас». При нажатии на кнопку автоматически проверяется наличие данной комбинации. Если у игрока нет заявленной комбинации (в таком случае обратный отсчет сразу отключается) или время кончилось, он автоматически пропускает ход. После того, как оставшиеся три игрока ответят, объявляются комбинации.
9. Если есть игрок, который хочет объявить Маджонг, на стол выкладывается собранная с помощью снесенной кости комбинация и игра заканчивается. Иначе, если есть игрок, собирающийся собрать Панг или Конг, соответствующая комбинация выкладывается на стол. Очередь переходит к данному игроку. Если игрок собрал Конг, ему предварительно выдается свободная кость из Стены. Иначе, если игрок справа от того, кто снес кость, заявил Чоу, соответствующая комбинация выкладывается на стол и наступает его очередь делать ход. Если никто из игроков не объявил никакую комбинацию, то следующий игрок (против часовой стрелки) получает кость со стены и делает ход (соответственно п.7).
10. Если у игрока есть 4 одинаковые кости и его очередь делать ход, он может нажать на кнопку «Конг». Тогда объявится закрытый Конг, комбинация будет выложена на стол. Игрок получит свободную кость со стены и отсчет времени для хода запустится заново. Аналогично, если у игрока есть открытый Панг, на руках есть 4-я такая же кость и его очередь делать ход, он может нажать на кнопку «Конг» и объявить открытый Конг. В таком случае 4-я кость выкладывается к 3-м на столе, игроку выдается свободная кость, и он делает ход. Также, если у игрока собраны комбинации для Маджонга и его очередь ходить, нажав на кнопку «Маджонг», он может завершить игру.
11. Если игроку в любой момент времени приходит со Стены цветок или сезон, он автоматически выкладывается на стол и игроку выдается свободная кость. На момент хода у игрока не может быть цветов и сезонов. Если в любой момент времени свободные кости заканчиваются, новые кости автоматически выкладываются на их место.
12. После объявления Маджонга кости всех игроков открываются. Автоматически высвечивается таблица со старыми очками, очками за раунд, информации о проигрыше и выигрыше другим игрокам, суммарными очками и новым рейтингом. Очки подсчитываются в соответствии с правилами подсчета очков (Приложение 2). Нажав на кнопку «Вид»/ «View» в таблице можно ее свернуть и посмотреть кости противников. С помощью такой же кнопки на верхней панели игрок может открыть вид сверху (необходимо нажать ту же кнопку для возвращения обратно). После нажатия на кнопку «Продолжить»/«Continue» все кости удаляются и игрок переходит в режим ожидания. Как только все игроки нажали на кнопку продолжения, ветра меняются, согласно правилам [3]. Запускается подтверждение готовности из п.4. Если после конца игры кто-то из игроков (не хост) покинул комнату, ветра не меняются, а суммарные очки сбрасываются к 2000. В таком случае, если зайдет новый игрок, ему необходимо будет подождать, пока оставшиеся игроки не нажмут кнопку продолжения. После этого аналогичным образом появляется всплывающее окно готовности из п.4.
13. Если в какой-то момент времени в Стене остается менее 7 пар костей, не считая свободных, то объявляется ничья. Игрокам высвечивается соответствующее сообщение, кости удаляются, запускается новый раунд.

### **3.2.2 Описания алгоритма проверки закрытых костей при объявлении Маджонга**

* В процессе игры игроки собирают комбинации. Открытые комбинации каждого игрока хранятся в списке openedTiles типа Combination в классе Player. Закрытые кости хранятся в этом же классе в списке playerTiles типа Tile. При заявке на объявление Маджонга у игрока часть костей (а иногда и все кости) закрыта. Для определения наличия Маджонга необходимо проверить все возможные варианты комбинаций закрытых костей.
* Когда игрок нажимает кнопку «Маджонг», запускается специальный метод CmdCheckMahJong() для проверки закрытых костей. В нем все закрытые кости разбиваются на списки по мастям (всего 5 списков – 3 для мастей + ветра + драконы). Затем по количеству костей в каждом списке определяется тот, в котором должна быть пара (число костей в таком списке должно давать остаток 2 при делении на 3). В этом списке просматриваются все возможные пары и запускается рекурсивный метод FillOneSuitCombinations, проверяющий, может ли список (без текущей пары) быть разбит на Чоу и Панги. (\*) Если разбиение найдено, то оставшиеся списки разбиваются на комбинации аналогичным методом.
* Метод FillOneSuitCombinations принимает на вход список костей и ссылку на список комбинаций. Метод с помощью 3-х вложенных циклов пробегает по всем тройкам костей из списка. Если какая-либо тройка образует Панг или Чоу, то создается новый список без этой тройки. Если рекурсивно вызванный от нового списка метод возвращает true, то найденная комбинация добавляется в список и метод возвращает true. Если в какой-то момент после очередной найденной тройки костей новый список имеет длину ноль, то он добавляет эту тройку в комбинации и возвращает true. Если все три цикла в методе полностью пройдены, значит, комбинаций не найдено – возвращается false.

\*Маджонг – набор из 4-х Пангов, Чоу или Конгов и пары, однако в закрытых комбинациях Конгов быть не может, иначе бы было несоответствие числа костей

\*Некоторые идеи для создания алгоритма взяты из источника [1] в списке литературы.

**Псевдокод метода CmdCheckMahJong**

//начало метода

if (!очередь делать ход && !очередь заявить комбинацию)

{

AbortMahJong();

return;

}

Закрытые кости=закрытые кости игрока;

if (очередь заявить комбинацию) Добавить в закрытые кости кость со стола;

Отсортировать закрытые кости ();

//0-бамбуки,1-доты,2-символы,3-ветра,4-драконы

Сеты костей по мастям;

Закрытые комбинации;

Номер сета с парой;

Сет костей с парой;

Заполнить сеты костей по мастям (ref сеты костей по мастям, закрытые кости);

//0-бамбуки,1-доты,2-символы,3-ветра,4-драконы

//проверка размера каждого сета

for (int номер сета = 0; номер сета < количество сетом по мастям; номер сета++)

{

Текущий сет;

if (размер текущего сета % 3 == 2 && размер сета с парой== 0)

{

Сет с парой=текущий сет;

Номер сета с парой=номер сета;

}

else if (размер текущего сета % 3 != 0)

{

AbortMahJong();

return;

}

}

//поиск пары в сете с парой

for (int номер текущей кости = 0; номер текущей кости < размер сета с парой - 1; номер текущей кости ++)

{

if (имя текущей кости == имя следующей кости)

{

Комбинации;

Сет без найденной пары;

if (размер сета без найденной пары == 0 || FillOneSuitCombinations (Сет без найденной пары, ref Комбинации))

{

Закрытые комбинации[номер сета с парой] = комбинации;

Добавить в Закрытые комбинации[номер сета с парой] найденную пару;

break;

}

}

if (разбиение не найдено)

{

AbortMahJong();

return;

}

}

//другие масти

for (int номер сета = 0; номер сета < количество сетом по мастям; номер сета++)

{

if (номер сета != номер сета с парой)

{

Комбинации;

if (размер текущего сета == 0 || FillOneSuitCombinations (текущий сет, ref Комбинации))

закрытые комбинации[номер сета] = Комбинации;

else

{

AbortMahJong();

return;

}

}

}

//ожидание других игроков и объявление Маджонга

//конец метода

**Псевдокод метода FillOneSuitCombinations**

bool FillOneSuitCombinations(сет одной масти, ref комбинации)

{

for (int номер первой кости = 0; номер первой кости < размер сета одной масти - 2; номер первой кости ++)

{

for (int номер второй кости = номер первой кости + 1; номер второй кости < размер сета одной масти - 1; номер второй кости++)

{

for (int номер третьей кости = номер второй кости + 1; номер третьей кости < размер сета одной масти; номер третьей кости ++)

{

Новый сет = сет одной масти;

if (кости образуют Панг)

{

Убрать кости из нового сета;

if (длина нового сета == 0)

{

Добавить в комбинации найденный Панг;

return true;

}else if (FillOneSuitCombinations(новый сет, ref комбинации))

{

Добавить в комбинации найденный Панг;

return true;

}

}

if (кости образуют Чоу)

{

Удалить кости из нового сета;

if (длина нового сета == 0)

{

Добавить в комбинации найденное Чоу;

return true;

}else if (FillOneSuitCombinations(новый сет, ref комбинации))

{

Добавить в комбинации найденное Чоу;

return true;

}

}

}

}

}

return false;

}//конец метода

## **3.3 Описание входных и выходных данных**

Программа считывает локально сохраненные настройки (звуковое сопровождение и язык) и имя игрока из файла playerPrefs.txt при его наличии и корректности.

Полученные данные обрабатываются в классе PlayerPrefs. При некорректных данных используются значения по умолчанию. Игрок может изменить данные в настройках игры, они будут сохранены в файл автоматически.

## **3.4. Описание и обоснование выбора состава технических средств**

Для корректной работы программы необходим ПК с операционной системой Windows 8 и выше, обладающий следующими минимальными характеристиками:

* минимум 180 MB оперативной памяти
* минимум 180 MB на жёстком диске

## **Описание и обоснование выбора состава программных средств**

Программа разработана под Windows, так как она является самой распространенной ОС.

Разработка велась с помощью межплатформенной среды разработки компьютерных игр Unity. В Unity реализована встроенная сетевая технология Unet, использовавшаяся при написании программы, а также реализован широкий спектр возможностей для создания компьютерных игр. Также это один из самых распространенных игровых движков.

Программа написана на языке C#, так как это являлось необходимым условием для выполнения курсовой работы.

# **4.ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ**

x

|  |  |
| --- | --- |
|  | [1] Algorithm to build a “hand” calculation for riichi mahjong game [Электронный ресурс] // cs.stackexchange.com: [сайт]. URL: https:/​/​cs.stackexchange.com/​questions/​64560/​algorithm-to-build-a-hand-calculation-for-riichi-mahjong-game (дата обращения: 16.Апрель.2019). |
|  | [2] Unity manual [Электронный ресурс] // Unity: [сайт]. URL: https:/​/​docs.unity3d.com/​2018.3/​Documentation/​Manual/ |
|  | [3] Китайские классические правила [Электронный ресурс] // mahjong.ru: [сайт]. [2007]. URL: http:/​/​mahjong.ru/​index.php?option=com\_content&view=article&id=7:2011-01-09-16-06-56&catid=2:-rules&Itemid=4 (дата обращения: 14.Апрель.2019). |

x

# **5. ПРИЛОЖЕНИЯ**

## **ПРИЛОЖЕНИЕ 1**

**ИСПОЛЬЗУЕМЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ**

Стена – квадрат из костей со сторонами 18 и высотой 2

Масть – один из трёх видов костей: Доты, Бамбук или Символы

Чоу – последовательность из костей одной масти

Панг – комбинация из 3-х одинаковых костей

Конг – комбинация из 4-х одинаковых костей

Открытая/закрытая комбинация – объявленная с помощью снесённой другим игроком кости/имеющаяся на руках комбинация

Маджонг – набор открытых и закрытых комбинаций, состоящий из 4-х Пангов, Конгов или Чоу и пары одинаковых костей

## **ПРИЛОЖЕНИЕ 2**

**Правила подсчета очков**

Правила подсчета максимально близки к правилам из источника [1]. Тем не менее, в правилах данной версии игры есть некоторые отличия, например отсутствие лимитированных комбинаций или другой способ подсчета очков за цветы и сезоны.

**Стоимость комбинаций**

|  |  |  |
| --- | --- | --- |
| **Комбинация** | **Открытая** | **Закрытая** |
| Чоу | 0 | 0 |
| Панг из 2-8 | 2 | 4 |
| Панг из 1, 9, ветров или драконов | 4 | 8 |
| Конг из 2-8 | 8 | 16 |
| Конг из 1, 9, ветров или драконов | 16 | 32 |
| Пара собственных ветров / преимущественных ветров/ драконов | 2 | 2 |
| За Маджонг | 20 | |
| За последнюю кость для Маджонга, взятую со Стены | 2 | |

**Удвоения**

|  |
| --- |
| **Общие** |
| Панг или Конг собственных ветров (открытый/закрытый) |
| Панг или Конг преимущественных ветров (открытый/закрытый) |
| Панг или Конг драконов |
| Чистая масть с ветрами и драконами |
| Только 1, 9, ветра и драконы |
| Чистая масть (удвоение трижды) |
| Только ветра и драконы (удвоение трижды) |
| **Только для выигравшего Маджонг** |
| Без Чоу |
| Мизер (только Чоу и пара) |
| Кость на Маджонг взята из свободных |
| Кость на Маджонг была ограблением открытого Конга |

**Цветы и сезоны\***

Каждый цветок и сезон принадлежит ветру. 1-Восток, 2-Юг, 3-Запад, 4-Север

-Не свой цветок или сезон – 2 очка

-Свой цветок или сезон – 4 очка

-Все 4 цветка или все 4 сезона – удвоение очков за цветы и сезоны

**\*Стоимость цветов и сезонов подсчитывается отдельно после всех основных расчетов и прибавляется к итоговой сумме**

В конце игры подсчитываются очки каждого игрока. Если победил Восток, то все остальные игроки выплачивают ему двойную сумму его очков. Если победитель другой ветер, то ему все игроки платят полную сумму, а Восток-двойную. Далее проигравшие рассчитываются между собой. Каждый игрок выплачивает/ получает разницу между его суммой и другим проигравшим. Восток выигрывает/выплачивает вдвойне. Все выигрыши/проигрыши относительно других игроков суммируются и прибавляются к старым очкам (первоначально – 2000).

## **ПРИЛОЖЕНИЕ 3**

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ**

|  |  |
| --- | --- |
| **Класс** | **Назначение** |
| AudioManager | Управление музыкой и звуками |
| BezierMove | Управляет всеми движениями костей, в том числе по кривой Безье |
| BuildWall | Скрипт, строящий Стену из костей |
| Chow | Комбинация «последовательность» |
| ClickHandler | Отслеживание кликов игрока по костям |
| Combination | Абстрактный класс, родитель Chow, Pung и Kong |
| East | Скрипт Восточного ветра |
| GameManager | Скрипт, управляющий основной игрой |
| GameMaster | Скрипт, хранящий состояние игры и собирающий необходимые данные об игроках перед стартом основной игры |
| HostGame | Отвечает за создание собственной игры |
| JoinGame | Отвечает за присоединение к существующей игре |
| Kong | Комбинация «4 одинаковых» |
| LocalizationManager | Управление языковой составляющей |
| LocalizedText | Управляет локализацией текста, к которому прикреплён |
| MahJong | Комбинация Маджонг, а также подсчет очков всех игроков в конце игры |
| MenuManager | Скрипт главного меню |
| NewNetworkManager | Перегрузка стандартного NetworkManager в Unity |
| North | Скрипт северного ветра |
| Pair | Пара фишек в комбинации на Маджонг |
| Player | Основной скрипт игрока |
| PlayerChat | Управление игровым чатом |
| PlayerPrefs | Считывание из файла и сохранение в файл пользовательских настроек |
| PlayerUI | Скрипт, отвечающий за интерфейс во время игры |
| Pung | Комбинация «3 одинаковых» |
| RoomListItem | Скрипт, отвечающий за кнопку матча в списке доступных игр |
| SetupPlayer | Отвечает за инициализацию данных игрока и их передачу GameMaster при присоединении к матчу |
| SliderDrag | Индикатор окончания движения слайдера звуковой настройки |
| South | Южный ветер |
| Table | «Кон», куда игроки сносят кости |
| Tile | Представляет игровую кость |
| TileName | Хранит название кости для всплывающей подсказки |
| ToolTip | Отвечает за всплывающую подсказку с названием кости |
| UiWinds | Управление текстовых компонент с названиями ветров и именами игроков |
| Wall | Представляет стену из фишек |
| WallPair | Представляет пару фишек (верхняя-нижняя) – в таком виде они лежат в стене |
| West | Западный ветер |
| Wind | Абстрактный класс ветра, родитель East, South, West, North |

## **ПРИЛОЖЕНИЕ 4.**

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ МЕТОДОВ, ПОЛЕЙ И СВОЙСТВ**

**Класс AudioManager**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| mixer | public | AudioMixer | Хранит ссылку на игровой аудиомиксер | |
| settingsPanel | private | GameObject | Игровая панель настроек | |
| soundSlider | private | Slider | Слайдер, отвечающий за громкость звука | |
| musicSlider | private | Slider | Слайдер, отвечающий за громкость музыки | |
| lobbyTheme | public | AudioClip | Музыкальная тема для меню и лобби | |
| gameTheme | public | AudioClip | Музыкальная тема для игры | |
| combSounds | public | List<AudioClip> | Звуки для комбинаций | |
| playingClip | public | AudioClip | Музыкальная тема, играющая в данный момент | |
| soundValue | public | Float | Значение ползунка громкости звука | |
| musicValue | public | float | Значение ползунка громкости музыки | |
| instance | private | AudioManager | Синглтон | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Awake | private | void |  | Контроль за единственностью instance |
| SetTheme | public | void |  | Смена музыкальной темы |
| SetReferences | public | void |  | Переопределяет ссылки на объекты интерфейса |
| SetSliders | public | void |  | Изменение положения слайдеров программно |
| Start | private | void |  | Определяет музыкальную тему, играющую при запуске |
| SetMusic | public | void | Float soundLevel | Изменение громкости музыки в соответствии с переданным значением |
| SetSound | public | void | Float soundLevel | Изменение громкости звуков в соответствии с переданным значением |

**Класс BezierMove**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| thirdRowHeight | private | float | Хранит значение по оси y третьего ряда Стены | |
| endPoint | public | Vector3 | Позиция точки, куда необходимо передвинуть кость | |
| endRotation | private | Quaternion | Угол вращения кости в конечной позиции | |
| p0,p1,p3 | private | Vector3 | Три точки для создания кривой Безье | |
| num | private | int | Индекс текущей точки в списке точек кривой Безье | |
| moving | public | bool | При moving==true кость начинает движение с помощью Update | |
| rotating | public | bool | При rotating==true кость начинает вращение с помощью Update | |
| speed | private | float | Скорость движения кости | |
| owner | public | Player | Новый владелец кости | |
| check | public | bool | Нужно ли проверять, движется ли кость | |
| pos | private | List<Vector3> | Список точек кривой Безье | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| OpenTile | public | void | float rotation | Открывает кость лицевой стороной к игроку |
| Move | public | void | Vector3 end, float rotation | Метод для движения кости со стены к игроку |
| WaitForMoving | private | IEnumerator | Float rotation | Запускает движение кости, по окончании движения переворачивает ее лицевой стороне к игроку |
| LieOut | public | void | Vector3 endPos, float rotation,bool closed | Выкладывает фишку на стол лицевой стороной вверх/вниз |
| GetBezierPosition | private | Vector3 | Float t | Вычисляет очередную точку кривой Безье |
| FillArray | private | void | Vector3 x0, Vector3 x3, float height | Заполняет список точек кривой Безье |
| StartMoveNewFreeTiles | public | void | Vector3 pos, bool isFirst, float r | Запуск корутины MoveNewFreeTiles |
| MoveNewFreeTiles | private | IEnumerator | Vector3 newPos, bool isFirst, float rotation | Кладет новые свободные кости на стену |
| MoveFreeTile | private | void | Float x, float y, float x, float rotation | Начинает движение новой свободной кости |
| SelectTile | public | void |  | Метод выбора фишки путем ее поднятия |
| DeselectTile | public | void |  | Опускает фишку обратно |
| ShowTile | public | void | Float rotation | Открывает кость в конце игры |
| FixedUpdate | private | void |  | Производит движение и поворот фишки каждый фрейм при необходимости |

**Класс BuildWall**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| startPosition1, startPosition2, startPositin3, startPosition3 | private | Vector3 | Координаты точек, с которых начинают строиться соответствующие части стены | |
| tilePrefab | public | GameObject | Префаб игровой кости | |
| tiles | public | List<List<WallPair> | Список всех костей стены | |
| coordinates | public | List<Vector2> | Хранит координаты спрайтов изображений на фишках на развертке | |
| availableTiles | private | List<Tile> | Список всех доступных для создания фишек | |
| availableTilesIndexes | private | List<int> | Список индексов, которые еще не были использованы | |
| indexes | public | List<int> | Список индексов всех костей в случайном порядке | |
| instance | public | BuildWall | Экземпляр класса (синглтон) | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Awake | private | void |  | Контролирует единственность instance, заполняет массив доступных индексов |
| FillCoordinates | private | void |  | Заполняет массив координат спрайтов |
| CreateTileVariants | private | void | GameObject tile | Создает все кости+ заполняет список доступных костей |
| Build | public | void | List<ind> | Строит Стену на основе переданного с сервера списка случайных индексов |
| FillIndexes | public | void |  | Заполняет список индексов случайным образом (для сервера) |
| Refresh | public | void |  | Сброс всех полей к исходному состоянию |
| CreateWall | public | void | Int num, GameObject tile, ref List<int> ind | Строит одну из четырех частей стены на основе переданных данных |

**Класс Chow**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Chow | public |  | Tile t1,Tile t2,Tile t3 | Вызывает базовый конструктор |
| CalculatePoints | public | int | String wind | Перегруженный метод подсчета очков за комбинацию |

**Класс Combination**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| doubling | public | int | Удвоение | |
| opened | public | bool | Открытая или закрытая комбинация | |
| tileList | public | List<Tile> | Ссылки на кости, составляющие комбинацию | |
| Свойства | | | | |
| Name | public | string | Название комбинации | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Combination | public |  | Tile tile1, Tile tile2, Tile tile3 | Базовый конструктор |
| Combination | public |  |  | Конструктор без параметров |
| CalculatePoints | public | int | String wind | Абстрактный метод подсчет очков за комбинацию |

**Класс ClickHandler**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| OnMouseDown | public | void |  | Регистрирует клик игрока по кости и вызывает соответствующий метод |

**Класс East**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| East | public |  | Player p | Конструкторы |
| East | public |  |  |
| MoveRightFreePosition | public | void | Ref Vector3 pos | Сдвигает позицию вправо |
| MoveLeftFreePosition | public | void | Ref Vector3 pos | Сдвигает позицию влево |
| MoveForwardPosition | public | void | Ref Vector3 pos | Сдвигает позицию к игроку |

**Класс GameManager**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| instance | public | GameManager | Синглтон | |
| bf | private | BinaryFormatter | Экземпляр форматтера | |
| wallIsBuilt | public | bool | Содержит информацию о том, построена ли Стена | |
| tilesAreGiven | public | bool | Информация о том, розданы ли фишки игрокам в начале игры | |
| waitForCombinations | private | bool | Ожидаются ли ответы про комбинации от игроков | |
| winds | public | List<Wind> | Список ветров в игре с ссылками на игроков | |
| majorWind | public | string | Преимущественный ветер | |
| stopTheGame | public | bool | Нужно ли остановить игру | |
| canStopTheGame | public | bool | Можно ли остановить игру | |
| winnerWind | Public | string | Ветер победителя | |
| numOfAnsweredPlayers | private | int | Число игроков, давших ответ про комбинации | |
| chowDeclarator,pungDeclarator,kongDeclarator, mahJongDeclarator | public | Player | Ссылки на игроков, которые хотят объявить соответствующие комбинации | |
| RefreshDel | public | void | Делегат для события RefreshEv | |
| RefreshEv | public | RefreshDel | События обновления полей всех классов в конце игры | |
| currentWind | private | int | Индекс ветра, который в данный момент делает ход | |
| Свойства | | | | |
| GameTable | public | Table | Экземпляр «кона» | |
| NumOfAnsweredPlayers | public | int | Свойство для numOfAnsweredPlayers | |
| CurrentWind | public | int | Соответствующее полю currentWind свойство | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Awake | private | void |  | Контролирует единственность instance, инициализирует данные при старте |
| Start | private | void |  | Добавление необходимых методов в событие RefreshEv |
| StartGame | public | void |  | Заполняет на сервере массив случайных индексов, отправляет команды клиентам для постройки стены |
| FixTilePositions | private | void |  | Подправляет позиции костей |
| DistributeTiles | public | void |  | Отправление команд для раздачи костей, сортировки, проверки на цветы у каждого игрока в начале игры |
| CheckAllPlayersForFlowers | private | void |  | Последовательно запускает метод проверки на наличие цветов и сезонов у каждого игрока из списка ветров |
| CheckForFlowers | public | void |  | Проверка костей одного игрока на наличие цветов и сезонов и соответствующая обработка информации |
| RpcBuildOnAllClients | private | void | String data | Десериализует строку с данными с сервера и на основании полученных данных строит стену для каждого клиента |
| Lighten | private | IEnumerator |  | Запуск корутины осветления экранов игроков |
| Build | private | IEnumerator |  | Ждет, пока экраны игроков затемнятся, и делает кости видимыми |
| RpcMakeTilesVisible | private | void |  | Делает стену видимой для всех клиентов |
| SortTiles | public | void |  | Запуск сортировки костей для всех игроков из списка |
| Prepare | private | void |  | Обновление полей и подготовка к заявкам игроков на комбинации |
| InvokeChange, ChangeTurn | public | void |  | Смена очереди хода |
| PrepareForCombinations | public | void |  | Вызов сортировки и Prepare перед началом комбинаций |
| BeginPlayerMove | private | void |  | «Выбирает» взятую со стены фишку и запускает отсчет времени для хода |
| DefineWind | private | int | Player player | Определяет ветер игрока, объявившего комбинацию |
| TargetSetTurn | private | void | NetworkConnection conn | Сообщает игроку о том, что его очередь ходить |
| Update | private | void |  | Следит за остановкой игры при необходимости, как только все игроки дадут ответ про комбинации,-вызывает их объявления |
| DeclareDraw | public | void |  | Объявление ничьи и завершение игры |
| OpenAllTiles | private | void |  | Открывает кости всех игроков в конце игры |
| FinishGame | public | void | MahJong mahjong | Подсчет очков всех игроков |
| SetScores | private | void |  | Передает данные для построения итоговой таблицы игрокам |
| DeclareCombination | public | void | String combination, int audioNum | Вызывает сообщение об объявлении комбинации и соответствующий звуковой эффект у всех игроков |
| RefreshAll | public | void |  | Обновление данных у всех игроков |
| RefreshWinds | public | void |  | Обновление свободных позиций всех ветров |
| OnRefresh | public | void |  | Вызывает событие RefreshEv |
| Refresh | public | void |  | Обновление полей GameManager |

**Класс GameMaster**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| playersToStart | private | uint | Константное значение игроков для начала игры | |
| instance | public | GameMaster | синглтон | |
| availableCameras | public | SynListInt | Содержит список камер, которые еще не заняты другими игроками | |
| lights | public | List<Gameobject> | Освещение для каждого из 4-х ветров | |
| roundsPlayed | private | int | Число сыгранных раундов | |
| playerCount | private | int | Количество игроков в комнате в данный момент | |
| readyPlayers | public | int | Количество игроков, подтвердивших готовность к старту | |
| readyToContinuePlayers | Public | int | Количество игроков, желающих продолжить игру после конца раунда | |
| playersToContinue | public | Int | Число игроков, которые должны нажать кнопку продолжения для нового раунда | |
| gameState | public | string | Хранит текущую фазу игры | |
| allCameras | public | List<Camera> | Список всех камер | |
| players | private | Dictionary<string,Player> | Словарь со всеми подключенными игроками | |
| Свойства | | | | |
| PlayerCount | public | int | Свойство для playerCount | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Awake | private | void |  | Контролирует единственность instance, инициализирует данные при старте |
| Start | public | void |  | Запуск инициализации доступных камер при старте при необходимости |
| Refresh | private | void |  | Обновление числа готовых игроков |
| InitCameras | private | void |  | Заполнение индексов доступных камер при старте |
| AddPlayer | public | void | String netID, int ord, string wind | Удаляет камеру игрока из доступных |
| RegisterPlayer | public | void | String netID, Player player | Добавление очередного игрока в словарь и обновление полей |
| AddLabel | public | void | SetupPlayer p | Добавление имен других игроков в интерфейс нового игрока |
| TestUnregister | public | void | String netID | Удаление очередного игрока из словаря и обновление полей |
| RpcRemovePlayerName | private | void | String wind | Удаляет имя покинувшего игру игрока из интерфейсов оставшихся |
| GetReady | private | void |  | Запуск корутины ожидания готовности для всех игроков |
| SetReady | private | void |  | Остановка корутины ожидания готовности для всех игроков |
| DarkenScreens | public | void |  | Запуск корутины затемнения экрана для всех игроков |
| LightenScreens | public | void |  | Запуск корутины осветления экрана для всех игроков |
| KickAFK | public | void |  | Отключение от игры тех игроков, которые не нажали кнопку готовности |
| DisableInfo | private | void |  | Отключение кнопки выхода и информации о количестве игроков при старте игры для всех клиентов |
| AssignWinds | private | void |  | Присваивает ссылки на игроков всех ветров списку winds в GameManager |
| TargetAddWind | private | void | NetworkConnection conn |  |
| ChangeWinds | private | void |  | Смена ветров и камер при начале нового раунда |
| Update | private | void |  | Контроль состояния игры через gameState каждый фрейм и запуск необходимых методов при смене состояния |

**Класс HostGame**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | | **Назначение** | |
| roomSize | private | | uint | | | Константный размер игровой комнаты | |
| networkManager | private | | NewNetworkManager | | | Ссылка на NetworkManager | |
| Свойства | | | | | | | |
| RoomName | public | | string | | | Название комнаты | |
| Методы | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | **Аргументы** | | **Назначение** |
| Start | | private | | void |  | | Инициализация полей, конфигурация NetworkManager |
| CreateRoom | | public | | void |  | | Создание нового матча |

**Класс JoinGame**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | | **Назначение** | |
| roomList | private | | List<GameObject> | | | Список с кнопками всех доступных комнат | |
| networkManager | public | | NewNetworkManager | | | Ссылка на NetworkManager | |
| roomPrefab | public | | GameObject | | | Префаб кнопки новой комнаты | |
| parentPanel | public | | GameObject | | | Ссылка на панель с кнопками комнат | |
| status | public | | Text | | | Текст информационного сообщения (загрузка, ошибка соединения и т.д.) | |
| hostPanel | public | | GameObject | | | Панель с сообщением о том, что хост покинул игру | |
| refresher | private | | IEnumerator | | | Ссылка на корутину обновления списка комнат | |
| Методы | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | **Аргументы** | | **Назначение** |
| Start | | private | | void |  | | Инициализация полей |
| EnableHostPanel | | public | | void |  | | Включение hostPanel |
| DisableHostPanel | | public | | void |  | | Выключение hostPanel |
| RefreshRoomList | | public | | void |  | | Запрос обновления информации о комнатах |
| OnMatchList | | private | | void | Bool success, string extendedInfo, List<MatchInfoSnapshot> matches | | Обновление списка доступных комнат |
| CreateRoomButton | | public | | void | MatchInfoSnapshot match | | Создание кнопки новой комнаты |
| ClearRoomList | | private | | void |  | | Очистка списка доступных комнат |
| JoinRoom | | public | | void | MatchInfoSnapshot match | | Присоединение к выбранной комнате |
| WaitForJoin | | private | | IEnumerator |  | | Ожидание соединения с комнатой и вывод сообщений |
| Refresher | | private | | IEnumerator |  | | Вызов обновления списка комнат через определенные промежутки времени |

**Класс Kong**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Kong | public |  | (1)Tile t1,Tile t2,Tile t3, Tile t4,bool openedComb  (2) Pung pung, Tile t4, bool openedComb | 2 конструктора |
| CalculatePoints | public | int | String wind | Перегруженный метод подсчета очков за комбинацию |

**Класс LocalizationManager**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| instance | public | LocalizationManager | Синглтон | |
| EngRu | private | Dictionary<string,string> | Словарь перевода с английского на русский | |
| RuEng | private | Dictionary<string,string> | Русско-английский словарь | |
| ChangeLanguageDel | public | void | Делегат смены языка | |
| ChangeEvent | public | ChangeLanguageDel | Событие смены языка | |
| label | public | Text | Текст DropDown с выбранным в данный момент языком | |
| settingsPanel | public | GameObject | Ссылка на панель настроек | |
| dropDown | public | DropDown | Ссылка на элемент DropDown с выбранным языком | |
| RuFont | public | Font | Шрифт для русского языка | |
| EngFont | public | Font | Текст для английского языка | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Awake | private | void |  | Контроль за единственностью instance и заполнение словарей |
| ChangeFont | public | void | Ref Text text | Смена шрифта при смене языка |
| SetReferences | public | void |  | Переопределяет ссылки на объекты интерфейса |
| GetLocalizedValue | public | string | String key | Получение перевода слова по ключу |
| OnLanguageChanged | public | void |  | Вызов события смены языка |

**Класс LocalizedText**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| key | public | string | Текстовый ключ для перевода | |
| lang | private | string | Язык ключа | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Start | private | void |  | Инициализация |
| ChangeLanguage | private | void |  | Смена текущего языка |
| OnDestroy | private | void |  | Удаление соответствующего метода из события смены языка при удалении объекта |
| ChangeText | private | void |  | Смена текста языка, если текущий язык не соответствует языку LocalizationManager |

**Класс MahJong**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| closedCombinations | public | List<List<Combination>> | Список закрытых комбинаций победителя | |
| openedCombinations | public | List<List<Combination>> | Список открытых комбинаций победителя | |
| flowers | public | List<Tile> | Список цветов и сезонов победителя | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| MahJong | public |  | List<List<Combination> comb | Конструктор со списком закрытых комбинаций |
| CalculateMahJongPoints | public | int | String wind, bool playerTurn, bool isFreeTile, int order | Подсчет очков победителя |
| GetFlowerPoints | private | int | List<Tile> flowers, int order | Подсчет очков за цветы и сезоны |
| CountTextOfWinnerDoubles | private | int |  | Подсчет удвоений для победителя |
| CountNotWinnerDoubles | public | int | List<Combination> opened, List<Tile> closed | Подсчет удвоений проигравших |
| CountNotWinnerScore | public | int | Ref List<combination> closedComb, List<Combination> opened, List<Tile> closed, List<Tile > flowers, string wind, int order | Подсчет очков проигравших |
| CalculatePoints | public | int | String wind | Перегруженный метод подсчета очков |

**Класс MenuManager**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Start | private | void |  | Ограничение фреймрейта |
| OnEnable | private | void |  | Добавление метода OnSceneLoaded к событию загрузки сцены |
| OnSceneLoaded | private | void | Scene scene, LoadSceneMode mode | Обработка загрузки разных сцен |
| OnDisable | private | void |  | Удаление метода OnSceneLoaded из события загрузки сцены |
| Quit | public | void |  | Выход из приложения |
| Play | public | void |  | Загрузка лобби из меню |
| LoadAsyncScene | private | IEnumerator |  | Асинхронная загрузка сцены |

**Класс NewNetworkManager**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| hostLeft | public | bool | Покинул ли хост игру | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| OnClientDisconnect | public | void | NetworkConnection conn | Индикатор того, что хост покинул игру + отсоединение игрока |
| OnServerDisconnect | public | void | NetworkConnection conn | Обработка отсоединение клиента от сервера (на сервере) |

**Класс North**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| North | public |  | Player p | Конструкторы |
| North | public |  |  |
| MoveRightFreePosition | public | void | Ref Vector3 pos | Сдвигает позицию вправо |
| MoveLeftFreePosition | public | void | Ref Vector3 pos | Сдвигает позицию влево |
| MoveForwardPosition | public | void | Ref Vector3 pos | Сдвигает позицию к игроку |

**Класс Pair**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Pair | public |  | Tile tile1, Tile tile2 | Конструктор |
| CalculatePoints | public | int | String wind | Подсчет очков за комбинацию |

**Класс Player**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | | **Назначение** |
| playerTiles | public | List<Tile> | | Закрытые кости игрока |
| openedTiles | public | List<Combination> | | Открытые комбинации игрока |
| closedCombinations | public | List<Combination> | | Закрытые комбинации победителя |
| flowers | public | List<Tile> | | Цветы и сезоны игрока |
| name | private | string | | Имя игрока |
| wind | public | string | | Название ветра |
| order | public | int | | Номер ветра |
| score | public | int | | Набранные в раунде очки |
| oldScore | public | int | | Общие очки за все раунды |
| tileToMove | public | Tile | | Кость, движение которой необходимо ослеживать |
| needToCheckMoving | public | bool | | Надо ли следить за движением кости |
| needFreeTile | private | bool | | Нужна ли игроку свободная кость |
| playerTurn | public | bool | | Очередь игрока делать снос |
| startedCoroutine | public | bool | | Запущена ли корутина для хода |
| turnForCombination | public | bool | | Очередь заявлять комбинацию |
| selectedTile | public | Tile | | Выбранная в данные момент кость |
| freeSpacePosition | private | Vector3 | | Положение первой «дыры» в костях после сноса/комбинации |
| freeSpaceIndex | private | int | | Индекс следующей после пробела кости |
| waitingCombination | public | Combination | | Заявленная, но еще не объявленная комбинация |
| isFreeTile | public | bool | | Для победителя: взята ли кость на победу из свободных |
| positionList | private | List<Vector3> | | 14 позиций закрытых костей игрока |
| Свойства | | | | |
| Camera | public | Camera | | Камера игрока |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Sort, RpcSort, SortTiles | public | void |  | SortTiles-непосредственная сортировка, Sort и RpcSort – её вызовы |
| RpcFix | public | void |  | Поправка позиций костей игрока на всех клиентах |
| RpcFillPositionList | public | void |  | Заполнение positionList для сортировки значениями с сервера |
| TargetSetCombinationTurn | public | void | NetworkConnection conn | Запуск времени на комбинацию у одного игрока |
| CmdSetPlayerTurn | private | void | Bool turn | Изменение очереди игрока делать ход на сервере |
| RpcTakeTableTile | private | void | Vector3 freePosition, float rotstion, bool closed | Взятие кости со стола для комбинации |
| RpcLieOutTile | public | void | Int index, Vector3 freePosition, float rotation, string array | Снос кости на стол или выкладывание цветка/сезона |
| CmdAddTileToPlayerArray, RpcAddTileToPlayerArray | public | void | Int currentWall, int currentPair, string tile | Добавление кости в список игрока на сервере и клиентах соответственно |
| CmdLieTileOnTable | public | void | Int index | Команда серверу выложить кость на стол |
| RpcSetLastTile | private | void | Int index | Инициализация последней кости на столе у клиентов |
| InvokeDelete | private | void |  | Вызов заполнения пробелов после комбинации или сноса |
| RpcDeleteFreeSpace | private | void | Vector3 position, int index | Непосредственно удаление «пробелов» |
| SelectTile | public | void | GameObject tile | Выбор кости (перед ходом) |
| TargetSelectLastTile | Public | void | NetworkConnection conn | Автоматический выбор последней кости перед ходом игрока |
| InvokeSelect | public | Void |  |
| CheckList | private | int | GameObject tile | Проверка костей на наличие кости по ссылке |
| CheckForTileInArray | private | int | String name | Проверка костей игрока на наличие кости по имени |
| Pass, CmdPass | public | void |  | Обработка нажатия кнопки пропуска хода |
| FindNumOfSimilarTiles | private | int | String name, ref int firstIndex | Возвращает количество костей игрока с данным именем |
| CheckChow, CmdCheckChow | Public/private | void |  | Проверка костей игрока на наличие Чоу |
| CheckPung, CmdCheckPung | Public/private | void |  | Проверка на наличие Панга |
| CheckKong, CmdCheckKong | Public/private | void |  | Проверка на наличие Конга |
| CheckMahJong/CmdCheckMahJong | Public/private | void |  | Проверка на наличие Маджонга |
| FillOneSuitCombinations | private | bool | List<Tile> suitSet, ref List<Combination> combinationList | Рекурсивная функция поиска комбинаций в списке одной масти |
| FillSuitSets | private | void | Ref List<List<Tile>> suitSets, List<Tile> tiles | Заполняет список костями одной масти |
| AbortMahJong | private | void |  | Отмена объявления Маджонга в случае его отсутствия |
| LieCombinationTiles | private | void | Int windPos | Выкладывание костей на Панг или Конг |
| RpcLieOutCombinationTile | private | void | Vector3 freePosition, float rotation, string name, bool closed | Выкладывание одной кости на комбинацию |
| DeclareClosedKong | public | void | Int firstIndex | Объявление закрытого Конга |
| DeclareKongFromPung | public | void | Pung pung, int index | Объявление Конга из Панга |
| DeclarePung | public | void | Int windPos | Объявление Панга |
| DeclareChow | public | void | Int windPos | Объявление Чоу |
| DeclareKong | public | void | Int windPos | Объявление Конга |
| DeclareMahJong | public | void | Int windPos | Объявление Маджонга |
| AskForFreeTile | private | Void |  | Серверный запрос на свободную кость |
| AskForMove | Private | void |  | Серверный запрос на начало хода |
| EnableCheck | private | void |  | Включает проверку на движение кости |
| RpcOpenTiles | public | Void | Float rotation | Открытие всех костей в конце раунда |
| GetNextFlower | private | void |  | Обработка следующего цветка или сезона |
| CheckForFlowers | public | bool |  | Проверка костей игрока на наличие цветов и сезонов |
| EnableToolTipsForTile | public | void |  | Включение всплывающей подсказки для кости |
| CmdPrepareForCombinations | private | void |  | Команда серверу начать ожидание комбинаций |
| CmdAddWind | public | void |  | Добавление игрока в серверный список ветров |
| CheckTileMoving | private | bool | Tile tile | Проверка кости на движение |
| Update | private | void |  | Контроль за движениями костей, цветами, свободными костями |
| RpcRefresh | public | void |  | Обновление полей класса |
| TargetRefresh | public | void | NetworkConnection conn | Обновление полей всех классов для конкретного игрока |

**Класс PlayerChat**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| chatTextPrefab | public | Text | Префаб сообщения в чате | |
| scrollRect | private | ScrollRect | Компонент ScrollRect окна чата | |
| chatScroll | public | GameObject | Окно чата | |
| enteredText | public | InputField | Ссылка на поле ввода | |
| content | public | GameObject | Окно содержимого чата, куда добавляются сообщения | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Start | public | void |  | инициализация |
| AddMessage | public | void |  | Добавление сообщения |
| CmdSendMessage,  RpcSendMessage | private | void | String name, string message | Отправка другим клиентам |
| ClearChat | public | void |  | Удаление всех сообщений |
| IncreaseImage | public | void | GameObject image | Увеличение картинки |
| DecreaseImage | public | void | GameObject image | Уменьшение картинки |
| Update | private | void |  | Отправка сообщения по нажатию Enter |

**Класс PlayerPrefs**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | | **Назначение** | |
| nameText | public | | InputField | | | Поле ввода имени | |
| playerName | private | | string | | | Имя игрока | |
| instance | public | | PlayerPrefs | | | Синглтон | |
| path | private | | string | | | Путь к файлу с настройками | |
| namePanel | private | | GameObject | | | Панель первого ввода имени | |
| Свойства | | | | | | | |
| Name | public | | string | | | Свойство для playerName | |
| Методы | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | **Аргументы** | | **Назначение** |
| Awake | | private | | void |  | | Инициализация полей |
| SetReferences | | public | | void |  | | Обновление ссылок на объекты |
| Start | | private | | void |  | | Считывание настроек в файле при запуске |
| EnterFirstName | | public | | void |  | | Обработка ввода первого имени |
| WriteFileData | | public | | void |  | | Запись настроек в файл |
| ChangeName | | public | | void |  | | Перезапись нового имени в файл |

**Класс PlayerUI**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| networkManager | private | NewNetworkManager | Синглтон NetworkManager | |
| player | private | Player | Ссылка на игрока | |
| infoText | private | Text | Информационный текст | |
| infoPanel | private | GameObject | Информационная панель | |
| startButton | private | GameObject | Кнопка готовности к началу раунда | |
| playerWind | public | Text | Ссылка на ветер игрока в интерфейсе | |
| QuitButton | public | GameObject | Ссылка на кнопку выхода | |
| countDown | Public | Text | Текст обратного отсчета | |
| declaration | Public | Text | Текст объявления комбинации | |
| combinationInfo | public | Text | Информация о ходе и отсутствии комбинации | |
| canvas | public | Canvas | Ссылка на канвас игрока | |
| starter | Private | IEnumerator | Корутина ожидания начала раунда | |
| CountDown | Public | IEnumerator | Корутина обратного отсчета для хода | |
| CombinationEnum | Public | IEnumerator | Корутина обратного отсчета для комбинации | |
| hider | private | IEnumerator | Корутина, прячущая информацию об объявленной другим игроком комбинации | |
| leave | Public | Bool | Должен ли игрок покинуть игру | |
| ready | public | bool | Готов ли игрок к началу игры | |
| blackScreen | public | GameObject | Черный экран | |
| toolTipText | public | Text | Текст подсказки | |
| chowPanel | Public | GameObject | Панель выбора Чоу | |
| scorePanel | Public | GameObject | Панель с итогом раунда | |
| leavePanel | public | GameObject | Панель с информацией о покинувшем игру игроке | |
| firstButton | Public | Button | Кнопки для соответствующего варианта Чоу | |
| secondButton | Public | Button |
| thirdButton | public | Button |
| firsctChow | Private | Combination | Три варианта Чоу | |
| secondChow | Private | Combination |
| thirdChow | private | Combination |
| mainCam | public | Camera | Ссылка на главную камеру | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Start | private | void |  | инициализация |
| Awake | private | void |  | Инициализация |
| Refresh | public | void |  | Обновление полей класса |
| TargetShowScores | public | void | NetworkConnection conn, int[] scores, int[] oldScores, string[] names, string winner | Собирает и высвечивает итоговую таблицу раунда |
| CmdSetScore | private | void | Int score | Обновляет итоговый счет у игроков |
| FillMatrix | private | void | Ref int[,] matrix, int[] scores, string winner | Заполнение таблицы выигрыша-проигрыша между игроками |
| TargetInfo | public | void | NetworkConnection conn,string info | Показывает информацию об очереди хода |
| ChangeInfo | public | void | String info |
| AskToContinueGame | Public | void |  | Оповещение сервера о готовности игрока к очередному раунду |
| CmdContinue | private | Void |  |
| ChangeCamera | public | void |  | Переключение между главной и основной камерами |
| TargetShowLeftPlayerInfo | public | void | NetworkConnection conn, string name, string wind | Показывает информацию о покинувшем комнату игроке |
| TargetShowInfo | public | void | NetworkConnection conn, string info | Запуск корутины показа информации о комбинации |
| HideInfo | private | IEnumerator |  | Показ информации об отказе в комбинации |
| StopHideInfo | ptivate | void |  | Остановка и сброс корутины |
| DeclareCombination | private | IEnumerator | String wind, string combination | Оповещение об объявленной другим игроком комбинации |
| TargetShowDeclaredCombination | public | void | NetworkConnection conn, string wind, string combination | Запуск корутины объявления комбинации и активация звукового сопровождения |
| WaitForStart | private | IEnumerator |  | Корутина обратного отсчета перед стартом раунда |
| TargetStopWaiting | Public | Void | NetworkConnection conn | Остановка корутины обратного отсчета перед стартом |
| StopWaiting | public | void |  |
| TargetStartWaiting | Public | void | NetworkConnection conn | Запуск корутины ожидания начала раунда |
| WaitForMove | private | IEnumerator |  | Корутина обратного отсчета для хода |
| LaunchWaitForMove | public | void |  | Запуск соответствующей корутины |
| TargetStopWaitingForMove | public | void | NetworkConnection conn | Остановка корутины обратного отсчета для хода |
| StopWaitingForMove | public | void |  |
| WaitForCombination | private | IEnumerator |  | Корутина ожидания комбинации |
| LaunchWaitForCombination | public | void |  | Запуск соответствующей корутины |
| TargetStopWaitingForCombination | public | void | NetworkConnection conn | Остановка корутины обратного отсчета для комбинации |
| StopWaitingForMove | public | void |  |
| CmdSetCombinationTurn | private | void | Bool turn | Смена очереди игрока заявлять комбинацию |
| DarkenTheScreen | private | IEnumerator |  | Затемнение экрана |
| TargetStartDarken | public | void | NetworkConnection conn | Запуск затемнения экрана |
| LightenScreen | private | IEnumerator |  | Осветление экрана |
| TargetStartLighten | public | void | NetworkConnection conn | Запуск осветления экрана |
| DropConnection | public | void |  | Отключение игрока от комнаты |
| Update | public | void |  | Отключение игрока по запросу сервера |
| CmdFailToStart | private | void |  | Сообщение серверу о том, что игрок не нажал кнопку старт перед началом раунда |
| RpcFailToStart | private | void |  |
| MarkReady | Public | void |  | Обработка готовности игрока начать раунд |
| CmdMarkReady | Public | void |  |
| RpcMarkReady | public | Void |  |
| TargetDisableInfoComponents | public | void | NetworkConnection conn | Отключает кнопку выхода после начала игры |
| CmdSetWallBuilt | private | void |  | Пометка того, что стена построена |
| MakeChowChoice | public | void | Combination first, Combination second, Combination Third | Передача информации о доступных Чоу игроку |
| GetChowSequence | private | void | Combination comb, ref string text | Заполнение текста кнопок Чоу |
| TargetGiveChowChoice | private | void | NetworkConnection conn, string text1, string text2,xtring text3 | Высвечивание панели Чоу у игрока |
| SelectFirstChow/CmdSelectFirst | Public/private | void |  | Обработка выбора игроком первого варианта Чоу |
| SelectSecondChow/CmdSelectSecond | Public/private | void |  | Обработка выбора игроком второго варианта Чоу |
| SelectThirdChow/CmdSelectThird | Public/private | void |  | Обработка выбора игроком третьего варианта Чоу |
| StopWaitForChow | private | void |  | Сообщение серверу о выборе игрока (Чоу) |
| DisableChowPanel | private | void |  | Отключение панели выбора Чоу |

**Класс Pung**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Методы | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | | **Аргументы** | **Назначение** |
| Pung | public | |  | | | (1)Tile t1,Tile t2,Tile t3  (2)Til1, tile2, Tile3 , bool opened | 2 конструктора |
| CalculatePoints | public | | int | | | String wind | Перегруженный метод подсчета очков за комбинацию |
| Поля | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | **Назначение** | | |
| additionalPosition | | public | | Vector3 | Позиция 4-ой фишки для объявления Конга из Панга | | |

**Класс RoomListItem**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | | **Назначение** | |
| match | | private | | MatchInfoSnapshot | | Информация о матче | |
| roomInfo | | public | | Text | | Информация о матче – имя и кол-во игроков | |
| joinRoomDelegate | | private | | JoinRoomDelegate | | Экземпляр делегата присоединения к комнате | |
| Методы | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | **Аргументы** | | **Назначение** |
| Setup | public | | void | | MatchInfoShapshot myMatch, JoinRoomDelegate joinRoomCallBack | | Инициализация полей |
| JoinRoom | public | | void | |  | | Присоединение к комнате |

**Класс SetupPlayer**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** | |
| player | public | Player | Ссылка на игрока | |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| Start | private | void |  | Инициализация полей |
| CmdAddPlayerName | private | void | String wind, string name | Добавление имени другого игрока в интерфейс |
| TargetAddPlayerName | public | void | NetworkConnection conn, string wind, string name |
| TargetChangeWind | public | void | NetworkConnection conn | Смена ветра в конце раунда |
| CmdChangeWind/RpcChangeWind | public | void | Int order, string wind | Смена ветра игрока на сервере и клиентах |
| RpcRefreshOldScore | public | void |  | Сброс общего счет к 2000 |
| SetOrder | private | void |  | Случайный выбор ветра на сервере при инициализации игрока |
| TargetSetOrder | private | void | NetworkConnection conn, int order | Инициализация полей игрока и создание камеры |
| AssignWind | private | void |  | Присваивание строкового названия ветра по его номеру |
| CmdAddPlayer/RpcAddPlayer | private | void | Int ord, string wind, string name | инициализация полей на клиентах и сервере |
| OnStartClient | public | void |  | Добавление игрока в словарь игроков |
| TargetDisconnect | private | void | NetworkConnection conn | Отсоединение игрока |

**Класс SliderDrag**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| OnPointerUp | public | void |  | Регистрирует окончание передвижения слайдера |

**Класс South**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| South | public |  | Player p | Конструкторы |
| South | public |  |  |
| MoveRightFreePosition | public | void | Ref Vector3 pos | Сдвигает позицию вправо |
| MoveLeftFreePosition | public | void | Ref Vector3 pos | Сдвигает позицию влево |
| MoveForwardPosition | public | void | Ref Vector3 pos | Сдвигает позицию к игроку |

**Класс Table**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | | **Назначение** | |
| lastTile | public | | Tile | | | Последняя кость, выложенная на стол | |
| startPosition | private | | Vector3 | | | Начальная позиция кости на столе | |
| currentPosition | private | | Vector3 | | | Текущая позиция кости на столе | |
| rotation | public | | float | | | Угол поворота кости на столе | |
| Свойства | | | | | | | |
| CurrentPosition | public | | Vector3 | | | Свойство для currentPosition | |
| Методы | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | **Аргументы** | | **Назначение** |
| Table | | public | |  |  | | Инициализация полей, Конструктор |
| SetReferences | | public | | void |  | | Обновление ссылок на объекты |
| MoveRightStartPosition | | public | | void |  | | Двигает текущую позицию кости вправо |
| MoveLeftStartPosition | | public | | void |  | | Двигает текущую позицию кости влево |

**Класс Tile**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | | **Назначение** | |
| name | public | | string | | | Название кости в коде | |
| tile | public | | GameObject | | | GameObject кости | |
| numOfTile | private | | int | | | Номер кости (для сортировки) | |
| isOwned | public | | bool | | | Принадлежит ли кость игроку | |
| Методы | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | **Аргументы** | | **Назначение** |
| Tile | | public | |  | GameObject tile, bool created, Vector2 coords | | Конструктор |
| CompareTo | | public | | int | Tile tileToCompare | | Перегруженный метод сравнение костей |
| MakeVisible | | public | | void |  | | Делает кость видимой |
| CreateName | | private | | void |  | | Присваивает имена кости |
| CreateFace | | private | | void | Vector2 coords | | Создание картинки на кости |
| ToString | | public | | string |  | | Перегруженный метод строкового представления |

**Класс TileName**

|  |  |  |  |
| --- | --- | --- | --- |
| Свойства | | | |
| Name | public | string | Название кости на английском для всплывающей подсказки |
| RuName | public | string | Название кости на русском для всплывающей подсказки |

**Класс ToolTip**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | | **Назначение** | |
| toolTipText | public | | Text | | | Текст всплывающей подсказки | |
| mousePos | private | | Vector3 | | | Позиция мыши | |
| needTip | private | | bool | | | Нужна ли подсказка | |
| Методы | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | **Аргументы** | | **Назначение** |
| OnMouseEnter | | private | | void |  | | Обработка наведения мыши на кость |
| OnMouseExit | | private | | void |  | | Обработка потери фокуса мыши |
| Start | | private | | void |  | | Инициализация полей |

**Класс UiWinds**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | | **Назначение** | |
| labels | public | | List<Text> | | | Ссылки на текстовые компоненты ветров в интерфейсе | |
| windLabels | private | | Dictionary<string, Text> | | | Словарь ветер-ссылка на текст | |
| Методы | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | **Аргументы** | | **Назначение** |
| AssignWinds | | public | | void |  | | В зависимости от ветра игрока определяет, где какой ветер |
| AssignLabels | | private | | void | String left, string up, string right | | Присваивание ветрам ссылок на текст |
| SetName | | public | | void | String wind, string name | | Добавляет соответствующему тексту имя игрока |
| RemoveName | | public | | void | String wind | | Удаляет имя покинувшего комнату игрока |

**Класс Wall**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | | **Назначение** | |
| tiles | public | | List<List<WallPair>> | | | Все кости в Стене | |
| freeTiles | public | | List<Tile> | | | Свободные кости | |
| instance | public | | Wall | | | Синглтон | |
| beginningWall | public | | Int | | | Номер части Стены, где будет начало разбора | |
| beginningPair | public | | int | | | Номер пары костей, где будет начало разбора | |
| currentWall | private | | Int | | | Номер части Стены, откуда берутся кости | |
| currentPair | Private | | int | | | Пара костей, являющаяся началом Стены | |
| freeTileIsMoving | public | | bool | | | Двигаются ли свободные кости в данный момент | |
| currentWind | private | | int | | | Текущий ветер | |
| freeTilesCount | private | | int | | | Число свободных костей | |
| checkFreeTiles | private | | bool | | | Нужно ли проверять движение свободных костей | |
| Методы | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | **Аргументы** | | **Назначение** |
| Awake | | private | | void |  | | Контроль единственности инстанса |
| Initialize | | public | | void | List<List<WallPair>> | | Инициализация костей Стены |
| AssignFreeTiles | | public | | void |  | | Определяет место разбора Стены |
| DistributeTiles | | public | | void |  | | Раздача костей игрокам |
| CheckForTheEndOfTheWall | | private | | void | Ref int currentWall, ref int currentPair | | Проверка конца очередной части Стены |
| RpcRefreshCurrentPair | | private | | void | Int wallNum, int pairNum | | Обновление текущей пары костей |
| TargetAddTileToPlayerArray | | private | | void | NetworkConnection conn, int currentWall, int currentPair, string tile | | Добавление кости в список костей игрока |
| GiveWallTile | | public | | void |  | | Выдает игроку кость на ход |
| CheckDraw | | private | | bool | Int wall, int pair | | Проверка ничьи |
| RpcMoveUpper | | private | | void | Int currentWall, int currentPair, Vector3 freePosition, float rotation | | Движение верхней кости пары |
| RpcMoveLower | | private | | void | Int currentWall, int currentPair, Vector3 freePosition, float rotation | | Движение нижней кости пары |
| RpcLieFreeTiles | | private | | void | Int wallNum, int restNum | | Обозначает новую пару свободных костей |
| CheckAndFill | | private | | void | Int wallNum, int num | | Кладет новую пару свободных костей |
| GiveFreeTile | | public | | void |  | | Отдает игроку свободную кость |
| RpcGiveFreeTile | | private | | void | Vector3 freePosition, float rotation | | Движение свободной кости к игроку |
| CheckFreeTiles | | private | | void |  | | Проверка необходимости положить новые свободные кости |
| Update | | private | | void |  | |
| Refresh | | public | | void |  | | Обновление полей класса |

**Класс WallPair**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | | **Назначение** | |
| upperTile | public | | Tile | | | Верхняя кость пары | |
| lowerTile | public | | Tile | | | Нижняя кость пары | |
| Методы | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | **Аргументы** | | **Назначение** |
| WallPair | | public | |  | Tile up, Tile down | | Конструкторы |
| WallPair | | public | |  |  | |

**Класс West**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| West | public |  | Player p | Конструкторы |
| West | public |  |  |
| MoveRightFreePosition | public | void | Ref Vector3 pos | Сдвигает позицию вправо |
| MoveLeftFreePosition | public | void | Ref Vector3 pos | Сдвигает позицию влево |
| MoveForwardPosition | public | void | Ref Vector3 pos | Сдвигает позицию к игроку |

**Класс Wind**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Поля | | | | | | | |
| **Имя** | **Доступ** | | **Тип** | | | **Назначение** | |
| freePosition | public | | Vector3 | | | Свободная позиция костей на руках | |
| freeOpenPosition | public | | Vector3 | | | Свободная позиция открытых комбинаций | |
| freeFlowerPosition | public | | Vector3 | | | Свободная позиция цветов и сезонов | |
| startPosition | public | | Vector3 | | | Стартовая позиция закрытых костей | |
| startOpenTilePosition | public | | Vector3 | | | Стартовая позиция комбинаций | |
| startFlowerPosition | public | | Vector3 | | | Стартовая позиция цветов и сезонов | |
| rotation | public | | float | | | Угол поворота | |
| Методы | | | | | | | |
| **Имя** | | **Доступ** | | **Тип** | **Аргументы** | | **Назначение** |
| CompareTo | | public | | int | Wind wind | | Перегруженный метод сравнения ветров по номеру |
| MoveRightFreePosition | | public | | void | Ref Vector3 pos | | Абстрактный метод, двигает позицию вправо |
| MoveLeftFreePosition | | public | | void | Ref Vector3 pos | | Абстрактный метод, двигает позицию влево |
| MoveForwardPosition | | public | | void | Ref Vector3 pos | | Абстрактный метод, двигает позицию на игрока |
| Refresh | | public | | void |  | | Обновление полей класса |

**ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированых |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |