Отчет по практической работе №17

Студента группы №81

Лазо Константина Олеговича

**Используя практическую работу №11-16 дополнить имеющуюся игру завершением при 2 условиях (смерти героя и при достижении правого края игрового поля). После завершения игры, игрок перемещает на страницу с таблицей рекордов.**

**Подробное описании страницы рекордов находится в WSR\_2\_MODULE.docx.  
  
Не забываем про правильную сортировку рекордов и кнопку «начать игру» после рекордов.**

**Результат:**

Вот так выглядит таблица рекордов:

****

По завершению игры отправляется ajax запрос, логика которого описана в /mechanics/ajax.js. Запрос отправляет данные текущего игрока на url ../php/register.js и получает данные обо всех игроках из базы данных

Логика сцены завершения игры находится в классе /scenes/end.js

Из важного можно отметить:

1. сортировку выводимых значений (Данные в таблице отсортированы по количеству убитых монстров по убыванию. Если несколько строк имеют одинаковое кол-во убитых монстров, то они сортируются между собой по времени по возрастанию. Если несколько строк имеют одинаковое количество убитых монстров и одинаковое время, то они занимают одинаковую позицию в рейтинге)

*/\*\**

*\* Сортирует сначала по убийствам, а затем по времени*

*\* @param {array} arr Массив объектов*

*\*/*

    sortScores(*arr*)

    {

*arr*.sort((*a*, *b*) => parseInt(*a*.score) == parseInt(*b*.score) ? (*a*.time > *b*.time ? 1 : -1) : *b*.score - *a*.score);

    }

1. После завершения игры открывается экран результатов, в котором формируется таблица с лучшими 10 игроками, если игрок не вошел в таблицу, то выводятся 9 лучших игроков, а в последней строке указывается место и результат игрока, только что завершившего игру. Текущий игрок выделен.

*/\*\**

*\* Рендерит результаты*

*\*/*

    renderScores()

    {

        var players = **this**.ajax.jsonResponse,

            currentPlayerCheck = 0, *// текущий ли игрок*

            place = 0, *// место игрока в рейтинге*

            number = 0; *// номер игрока вообще*

**this**.sortScores(players);

        for(let i in players)

        {

            var player = players[i];

*// если время предыдущего игрока равно текущему, то занимаемое место не меняется*

            place += ((i > 0) && (players[i - 1].time) == player.time ? 0 : 1);

            number++;

*// если текущий игрок, то он выделяется в таблице рейтинга жирным шрифтом*

            if (**this**.player.nickname == player.username && **this**.player.kills == player.score && **this**.game.hud.time == player.time && currentPlayerCheck == 0)

            {

                if(number > 10)

                {

                    $('#scores tr:last').remove();

                    number = 10;

                }

                var currentPlayer = true;

                currentPlayerCheck = 1;

            }

            else

            {

                var currentPlayer = false

            }

            var score = (currentPlayer ? '<tr style="font-weight: bolder; background: #DCDCDC">' : '<tr>') +

                        '<td>'

                        + place +

                        '</td> <td>'

                        + player.username +

                        '</td> <td>'

                        + player.score +

                        '</td> <td>'

                        + player.time +

                        '</td> </tr>'

                        + (currentPlayer ? '</b>' : '');

            if(number <= 10)

            {

                $('#scores').append(score);

            }

        }

    }

1. Кнопка “Начать заново” и логика респауна в классе игрока и классах сцен

*/\*\**

*\* Кнопка "начать сначала"*

*\*/*

    startAgain()

    {

        let self = **this**;

        let submit = $('#' + **this**.buttonStartAgainId);

        switch(**this**.ajax.jsonResponse)

        {

            case '':

                submit[0].style.background = '';

                submit[0].disabled = 'true';

            break

            default:

                submit[0].disabled = '';

                submit[0].style.background = 'green';

                submit[0].onclick = function(*e*)

                {

*e*.preventDefault();

                    self.scene.hidden = 'true';

                    self.finish(Scene.START\_GAME);

                    self.player.respawn();

                }

            break

        }

        submit[0].onmouseout = function() {

**this**.style.background = '';

        }

    }

И прочие мелкие многочисленные изменения. Более подробно о разных изменениях можно почитать в репозитории на гитхабе.