**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ**

**И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

(СПбГУТ)

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ (**ИТПИ**)

КАФЕДРА ПРОГРАММНОЙ ИНЖЕНЕРИИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ (ПИ И ВТ)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа №2**

по дисциплине «Разработка фронтенд-приложений управления телекоммуникациями»

Тема «Основы JavaScript»

Вариант 24

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент гр. | ИКПИ-23 |  | 02.12.2024 | Харлова А.А. |
| Преподаватель |  |  |  | Белая Т.И. |

г.Санкт-Петербург

2024 г.

# ПОСТАНОВКА ЗАДАЧИ

В настоящей лабораторной работе следует реализовать игру на точность "Попади в цель". Необходимо:

1. Создать мишени. Разместить их на экране.
2. Реализовать управление стрельбой по мишеням.
3. Добавить логику для проверки попадания в мишень и учёта очков.
4. Отображать счёт и уровни сложности.

**РАЗРАБОТКА АЛГОРИТМА**

Алгоритм написания игры на точность "Попади в цель" разделить на несколько шагов. Этот алгоритм будет включать в себя использование HTML, CSS и JavaScript.

Подзадачи:

1. Создание HTML структуры (lab2.html), содержащей основные блоки веб-страницы.
2. Добавление CSS стилей. Создание CSS файл (styles.css) для базовых стилей и начальных состояний текста.
3. Написание и подключение JavaScript для анимации (handlers.js).

Реализация подзадачи 1.

Создаём шаблон файла HTML5 и подключаем JS и CSS-файлы.

Реализация подзадачи 2.

Создаем CSS-файл стилей «styles.css».

Реализация подзадачи 3.

Для удобства весь код JS был разделен на четыре файла «parametres.js», «functions.js», «handlers.js», «updaters.js».

**«parametres.js»**

В файле «parametres.js» определяем основные настройки оформления и логики игры.

Для корректного отображения элементов на экранах любого размера получаем параметры экрана пользователя.

*/\*------------------ Common ------------------\*/***const *screenHeight*** = ***window***.**innerHeight**;  
**const *screenWidth*** = ***window***.**innerWidth**;  
**const *currentPos*** = [];

Определяем настройки оформления (цвета анимаций и размер текста).

*/\*------------------ DecorationParams ------------------\*/***const *DEFAULT\_TEXT\_COLOR*** = **'black'**;  
**const *triggerColor*** = [**'#00B64F'**, **'#EA0037'**];  
**const *mainColors*** = [**'#23405F'**]  
**const *scoreData*** = {  
 **'100'**: {**'fontSize'**: 2.7, **'fontWeight'**: **'bolder'**, **'color'**: ***mainColors***[0]},  
 **'200'**: {**'fontSize'**: 4, **'fontWeight'**: **'bolder'**, **'color'**: ***mainColors***[0]},  
 **'300'**: {**'fontSize'**: 5, **'fontWeight'**: **'bolder'**, **'color'**: ***mainColors***[0]},  
 **'400'**: {**'fontSize'**: 6, **'fontWeight'**: **'bolder'**, **'color'**: ***triggerColor***[0]},  
 **'500'**: {**'fontSize'**: 7.5, **'fontWeight'**: **'bolder'**, **'color'**: ***triggerColor***[0]},  
 **'-100'**: {**'fontSize'**: 6, **'fontWeight'**: **'bolder'**, **'color'**: ***triggerColor***[1]}  
};

Задаём логические параметры игры. Включают в себя счетчики, скорость объектов, чувствительность управления клавиатурой.  
*/\*------------------ GameInit ------------------\*/***let *scoreCounter*** = 0;  
**let *currentLevel*** = 1;  
**let *misclicksCounter*** = 0;  
**let *hitsCounter*** = 0;  
**let *targetCounter*** = 0;  
  
**let *timer*** = 2400;  
**let *newTargetId*** = 0;  
**let *targetTimeOuts*** = {};  
  
*//GameParams***const *levelUpdateValue*** = 1.5;  
**const *stepAimMover*** = 50;  
  
*//TimeParams***const *timerStep*** = 350;  
**const *maxSpeedTimerValue*** = 500;  
**const *minSpeedTimerValue*** = 2400;

**«functions.js»**

Создаём файл «functions.js», содержащий необходимые независимые функции.

Математическая функция для генерирования рандомного числа в определнном интервале.

*/\*------------------ MATH ------------------\*/***function** *randomIntFromInterval*(min, max) { *// min and max include* **return *Math***.floor(***Math***.random() \* (max - min + 1)) + min;  
}

Функция, отвечающая за перемещение мишени раз в определенный интервал времени.  
  
*/\*------------------ MOVER'S ------------------\*/***function** *targetMover*(elem) {  
 elem.**style**.**left** = *randomIntFromInterval*(0.1 \* ***screenWidth***, 0.8 \* ***screenWidth***) + **'px'**;  
 elem.**style**.**top** = *randomIntFromInterval*(0.1 \* ***screenHeight***, 0.8 \* ***screenHeight***) + **'px'**;  
  
 **if** (elem.**id in *targetTimeOuts***) {  
 *clearTimeout*(***targetTimeOuts***[elem.**id**]);  
 }  
 ***targetTimeOuts***[elem.**id**] = *setTimeout*(*targetMover*, ***timer***, elem);  
}

Данный блок функций необходим для создания плавных анимаций элементов.

Функция, плавно изменяющая стиль объекта, а затем удаляющая элемент со страницы.

*/\*------------------ NICE ANIMATIONS ------------------\*/***function** *animateAndRemove*(elem, className) {  
 elem.**classList**.add(className);  
 elem.addEventListener(**'transitionend'**, **function** *templeRemover*() {  
 **this**.removeEventListener(**'transitionend'**, *templeRemover*);  
 elem.**parentElement**.removeChild(elem);  
 });  
}

Функция добавления нового всплывающего элемента, отображающего количество заработанных при выстреле очков.

**function** *addNewPopUpScore*(data, style, x, y) {  
 **let** newElem = ***document***.createElement(**'text'**);  
 newElem.**innerHTML** = data;  
 newElem.**classList**.add(**'popUpScore'**);  
 newElem.**style**.**left** = x + **'px'**;  
 newElem.**style**.**top** = y + **'px'**;  
 newElem.**style**.**fontSize** = style[**'fontSize'**] + **'vh'**;  
 newElem.**style**.**fontWeight** = style[**'fontWeight'**];  
 newElem.**style**.**color** = style[**'color'**];  
  
 ***document***.**body**.append(newElem);  
 ***window***.setTimeout(*animateAndRemove*.bind(**this**, newElem, **'fade-out'**), 100);  
}

Функция добавления нового элемента при промахе пользователя в определённые координаты страницы.

**function** *addMissclick*(x, y) {  
 **let** newElem = ***document***.createElement(**'div'**);  
 newElem.**classList**.add(**'missclickPoint'**);  
 newElem.**style**.**left** = x + **'px'**;  
 newElem.**style**.**top** = y + **'px'**;  
 ***document***.**body**.append(newElem);  
 ***console***.log(**"its OK"**);  
 ***window***.setTimeout(*animateAndRemove*.bind(**this**, newElem, **'disappear'**), 10);  
}

Функция, отвечающая за анимацию текста (изменение цвета и размера).

**function** *trigger*(elem, color = ***DEFAULT\_TEXT\_COLOR***) {  
 elem.**style**.**transform** = **'scale('** + 1.2.toString() + **')'**;  
 elem.**style**.**color** = color;  
 elem.addEventListener(**'transitionend'**, **function** () {  
 elem.**style**.**transform** = **'scale(1)'**;  
 elem.**style**.**color** = ***DEFAULT\_TEXT\_COLOR***;  
 }  
 );  
}

Функция, отвечающая за анимацию курсора мыши при клике на экран.

**function** *mouseTrigger*(value, deg) {  
 ***aimElem***.**style**.**transform** = **'scale('** + value + **')'** + **' '** + **'rotate('** + deg + **')'**;  
 ***aimElem***.addEventListener(**'transitionend'**, **function** () {  
 ***aimElem***.**style**.**transform** = **'scale(1)'**;  
 }  
 );  
}

Функция, отвечающая за анимацию блока-мишени при попадании пользователем.

**function** *targetTrigger*(elem) {  
 elem.**style**.**transform** = **'scale('** + 1.2.toString() + **')'**;  
 elem.**style**.**opacity** = **'0.1'**;  
 elem.addEventListener(**'transitionend'**, **function** () {  
 elem.**style**.**transform** = **'scale(1)'**;  
 elem.**style**.**opacity** = **'1'**;  
 }  
 );  
}

**«handlers.js»**

Создаём файл «handlers.js». Данный файл включает в себя основные обработчики событий страницы.

Ссылки на часто используемые HTML объекты записываем в переменные JS.

*/\*------------------ Varibles ------------------\*/***const *aimElem*** = ***document***.getElementById(**'aimWrapper'**);  
**let *leftAimPos*** = ***aimElem***.getBoundingClientRect().**left**;  
**let *topAimPos*** = ***aimElem***.getBoundingClientRect().**top**;

Обработчик нажатия клавиш для управления клавиатурой. Включает в себя проверку выхода курсора мыши за экран и его циклический перенос.  
  
*/\*------------------ Click/Keydown Handlers ------------------\*/****window***.addEventListener(**'load'**, () => {  
 ***window***.addEventListener(**'keydown'**, event => {  
 **switch** (event.**key**) {  
 **case 'ArrowLeft'**:  
 **case 'a'**:  
 **case 'ф'**:  
 ***leftAimPos*** -= ***stepAimMover***;  
 **if** (***leftAimPos*** <= 0) {  
 ***leftAimPos*** = ***screenWidth*** - ***aimElem***.**clientWidth**;  
 }  
 **break  
  
 case 'ArrowRight'**:  
 **case 'd'**:  
 **case 'в'**:  
 ***leftAimPos*** += ***stepAimMover***;  
 **if** (***leftAimPos*** >= ***screenWidth*** - ***aimElem***.**clientWidth** / 2) {  
 ***leftAimPos*** = 0;  
 }  
 **break  
  
 case 'ArrowDown'**:  
 **case 's'**:  
 **case 'ы'**:  
 ***topAimPos*** += ***stepAimMover***;  
 **if** (***topAimPos*** >= ***screenHeight*** - ***aimElem***.**clientHeight** / 2) {  
 ***topAimPos*** = 0;  
 }  
 **break  
  
 case 'ArrowUp'**:  
 **case 'w'**:  
 **case 'ц'**:  
 ***topAimPos*** -= ***stepAimMover***;  
 **if** (***topAimPos*** <= 0) {  
 ***topAimPos*** = ***screenHeight*** - ***aimElem***.**clientHeight**;  
 }  
 **break  
  
 case ' '**:  
 **const** x = ***aimElem***.getBoundingClientRect().**x** + ***aimElem***.**clientWidth** / 2; *// получаем координату X мыши* **const** y = ***aimElem***.getBoundingClientRect().**y** + ***aimElem***.**clientHeight** / 2; *// получаем координату Y мыши;* **const** mouseClickEvent = **new** *MouseEvent*(**'click'**, {  
 **'view'**: ***window***,  
 **'bubbles'**: **true**,  
 **'cancelable'**: **true**,  
 **'clientX'**: x,  
 **'clientY'**: y  
 });  
 ***document***.elementFromPoint(x, y).dispatchEvent(mouseClickEvent);  
 **break** }  
 ***aimElem***.**style**.**left** = ***leftAimPos*** + **'px'**;  
 ***aimElem***.**style**.**top** = ***topAimPos*** + **'px'**;  
 });  
});

Функция, обрабатывающая событие «click». Создает всплывающие блоки отображения заработанных очков, обновляет игровые параметры, вызывает исполнение нужных анимаций.  
  
**function** *targetClickHandler*(event, index) {  
 **const** x = event.**clientX**;  
 **const** y = event.**clientY**;  
 event.stopPropagation();  
  
 **if** (index === 5) {  
 *addMissclick*(event.**clientX**, event.**clientY**);  
 *misclicksUpdater*();  
 *mouseTrigger*(**'0.8'**, **'-90deg'**);  
 } **else** {  
 **let** elem = event.**target**;  
 **while** (!(elem.**classList**.contains(**'target-wrapper'**))) {  
  
 elem = elem.**parentElement**;  
 }  
 *mouseTrigger*(**'1.4'**, **'90deg'**);  
 *targetMover*(elem);  
 *targetTrigger*(elem);  
 *hitsUpdater*();  
 }  
 **let** entry = ***Object***.keys(***scoreData***)[index];  
 *addNewPopUpScore*(entry, ***scoreData***[entry], x, y);  
 *scoreUpdater*(*parseInt*(entry));  
}

Обработчик «мискликов».

***document***.getElementById(**'mainArea'**).addEventListener(**'click'**,  
 event => *targetClickHandler*(event, 5));

Обработчик перемещения указателя мыши для персонализированного курсора.

***window***.addEventListener(**'mousemove'**, **function** (event) {  
 ***document***.getElementById(**'aimWrapper'**).**style**.**left** = event.**clientX** - 0.03 \* ***Math***.min(***screenHeight***, ***screenWidth***) + **'px'**;  
 ***document***.getElementById(**'aimWrapper'**).**style**.**top** = event.**clientY** - 0.03 \* ***Math***.min(***screenHeight***, ***screenWidth***) + **'px'**;  
})

**«updaters.js»**

Создаём файл «updaters.js». Данный файл содержит функции для обработки изменений и их отображения на экране пользователя в ходе игры.

Создание JS переменных часто используемых HTML объектов.

*/\*------------------ Varibles ------------------\*/***let *scoreLabel*** = ***document***.getElementById(**'scoreCounterLabel'**);  
**let *levelLabel*** = ***document***.getElementById(**'levelCounterLabel'**);

Функции для обновления счётчиков в ходе игры.

*/\*------------------ GAME LOGIC ------------------\*/***function** *scoreUpdater*(value) {  
 **if** (value > 0) {  
 *trigger*(***scoreLabel***, ***triggerColor***[0]);  
 } **else** {  
 *trigger*(***scoreLabel***, ***triggerColor***[1]);  
 }  
 ***scoreCounter*** += value;  
 ***scoreLabel***.**innerHTML** = **'Score: '** + **scoreCounter**.toString();  
 *levelUpdater*();  
}  
  
**function** *levelUpdater*() {  
 **if** (***currentLevel*** \* ***levelUpdateValue*** \* 1000 <= ***scoreCounter***) {  
 *trigger*(***levelLabel***, ***triggerColor***[0]);  
 ***currentLevel*** += 1;  
 ***timer*** = ***Math***.max(***timer*** - ***timerStep***, ***maxSpeedTimerValue***);  
  
 } **else if** ((***currentLevel*** - 1) \* ***levelUpdateValue*** \* 1000 > ***scoreCounter***) {  
 **if** (***currentLevel*** !== 1) {  
 *trigger*(***levelLabel***, ***triggerColor***[1]);  
 ***currentLevel*** -= 1;  
 ***timer*** = ***Math***.min(***timer*** + ***timerStep***, ***minSpeedTimerValue***);  
 }  
 }  
 ***levelLabel***.**innerHTML** = **'Lvl: '** + ***currentLevel***.toString();  
}  
  
**function** *misclicksUpdater*() {  
 ***misclicksCounter*** += 1;  
 ***document***.getElementById(**'misclickStatLabel'**).**innerHTML** = **`Misclicks:** ${**misclicksCounter**}**`**;  
}  
  
**function** *hitsUpdater*() {  
 ***hitsCounter*** += 1;  
 ***document***.getElementById(**'hitStatLabel'**).**innerHTML** = **`Hits:** ${**hitsCounter**}**`**;  
}  
  
**function** *targetUpdater*(value) {  
 ***targetCounter*** += value;  
 ***document***.getElementById(**'targetCounterLabel'**).**innerHTML** = **`Targets:** ${**targetCounter**}**`**;  
}

Функция добавления нового HTML объекта (мишени).

*/\*------------------ BUTTON HANDLERS ------------------\*/***function** *addTarget*() {  
 **let** newElem = ***document***.createElement(**'div'**);  
 newElem.**classList**.add(**'target-wrapper'**);  
 newElem.**id** = **`trgN**${**newTargetId**}**`**;  
 ***newTargetId*** += 1;  
 newElem.**innerHTML** =  
 **' <div class="target trg1">\n'** +  
 **' <div class="target trg2">\n'** +  
 **' <div class="target trg3">\n'** +  
 **' <div class="target trg4">\n'** +  
 **' <div class="target trg5">\n'** +  
 **' </div>\n'** +  
 **' </div>\n'** +  
 **' </div>\n'** +  
 **' </div>\n'** +  
 **' </div>\n'** +  
 **'</div>'  
 *document***.getElementById(**'mainArea'**).appendChild(newElem);  
 **for** (**let** i = 1; i < 6; i++) {  
 newElem.querySelector(**`.trg**${i}**`**).addEventListener(**'click'**, event => *targetClickHandler*(event, i - 1));  
 }  
 *targetMover*(newElem);  
 *targetUpdater*(1);  
}

Функция удаления HTML объекта (мишени) с обработкой недопустимых случаев.

**function** *deleteTarget*() {  
 **if** (***targetCounter*** > 1) {  
 **let** elem = ***document***.querySelector(**'.trg1'**);  
 elem.removeEventListener(**'click'**, *targetClickHandler*);  
 elem.remove();  
 *targetUpdater*(-1);  
 } **else** {  
 *alert*(**'Число мишеней не может быть меньше'**);  
 }  
}

**ПРИЛОЖЕНИЕ**

Файл «lab2.html».

<!DOCTYPE **html**>  
<**html lang="en"**>  
<**head**>  
 <**meta charset="UTF-8"**>  
 <**title**>Shoot!</**title**>  
 <**link rel="stylesheet" type="text/css" href="styles.css"**/>  
</**head**>  
<**body**>  
<**script src="parametres.js"**></**script**>  
  
<**div id="header"**>  
 <**h3 id="scoreCounterLabel"**>Score: 0</**h3**>  
 <**h1 id="mainHeader"**>Shoooot!</**h1**>  
 <**h3 id="levelCounterLabel"**>Lvl: 1</**h3**>  
</**div**>  
  
<**div id="aimWrapper"**>  
 <**div id="aim"**></**div**>  
</**div**>  
  
<**div id="mainArea"**></**div**>  
</**body**>  
  
<**footer**>  
 <**div class="statData" id="statDataBlock"**>  
 <**h4 id="hitStatLabel"**>Hits: 0</**h4**>  
 <**h4 id="misclickStatLabel"**>Misclicks: 0</**h4**>  
 </**div**>  
  
 <**div class="statData" id="statTargetsBlock"**>  
 <**button class="targetBtns" id="delTargetBtn" onclick="***deleteTarget*()**"**></**button**>  
 <**h2 id="targetCounterLabel"**>Targets: 1</**h2**>  
 <**button class="targetBtns" id="addTargetBtn" onclick="***addTarget*()**"**></**button**>  
 </**div**>  
  
 <**div class="statData" id="creator"**>  
 <**h4**>IKPI-23</**h4**>  
 <**h4**>Kharlova A.A.</**h4**>  
 </**div**>  
</**footer**>  
  
<**script src="functions.js"**></**script**>  
<**script src="handlers.js"**></**script**>  
<**script src="updaters.js"**></**script**>  
  
<**script**>  
 *addTarget*();  
</**script**>  
  
</**html**>

Файл «parametres.js».

*/\*------------------ Common ------------------\*/***const *screenHeight*** = ***window***.**innerHeight**;  
**const *screenWidth*** = ***window***.**innerWidth**;  
**const *currentPos*** = [];  
  
*/\*------------------ DecorationParams ------------------\*/***const *DEFAULT\_TEXT\_COLOR*** = **'black'**;  
**const *triggerColor*** = [**'#00B64F'**, **'#EA0037'**];  
**const *mainColors*** = [**'#23405F'**]  
**const *scoreData*** = {  
 **'100'**: {**'fontSize'**: 2.7, **'fontWeight'**: **'bolder'**, **'color'**: ***mainColors***[0]},  
 **'200'**: {**'fontSize'**: 4, **'fontWeight'**: **'bolder'**, **'color'**: ***mainColors***[0]},  
 **'300'**: {**'fontSize'**: 5, **'fontWeight'**: **'bolder'**, **'color'**: ***mainColors***[0]},  
 **'400'**: {**'fontSize'**: 6, **'fontWeight'**: **'bolder'**, **'color'**: ***triggerColor***[0]},  
 **'500'**: {**'fontSize'**: 7.5, **'fontWeight'**: **'bolder'**, **'color'**: ***triggerColor***[0]},  
 **'-100'**: {**'fontSize'**: 6, **'fontWeight'**: **'bolder'**, **'color'**: ***triggerColor***[1]}  
};  
  
*/\*------------------ GameInit ------------------\*/***let *scoreCounter*** = 0;  
**let *currentLevel*** = 1;  
**let *misclicksCounter*** = 0;  
**let *hitsCounter*** = 0;  
**let *targetCounter*** = 0;  
  
**let *timer*** = 2400;  
**let *newTargetId*** = 0;  
**let *targetTimeOuts*** = {};  
  
*//GameParams***const *levelUpdateValue*** = 1.5;  
**const *stepAimMover*** = 50;  
  
*//TimeParams***const *timerStep*** = 350;  
**const *maxSpeedTimerValue*** = 500;  
**const *minSpeedTimerValue*** = 2400;

Файл «functions.js».

*/\*------------------ MATH ------------------\*/***function** *randomIntFromInterval*(min, max) { *// min and max include* **return *Math***.floor(***Math***.random() \* (max - min + 1)) + min;  
}  
  
*/\*------------------ MOVER'S ------------------\*/***function** *targetMover*(elem) {  
 elem.**style**.**left** = *randomIntFromInterval*(0.1 \* ***screenWidth***, 0.8 \* ***screenWidth***) + **'px'**;  
 elem.**style**.**top** = *randomIntFromInterval*(0.1 \* ***screenHeight***, 0.8 \* ***screenHeight***) + **'px'**;  
  
 **if** (elem.**id in *targetTimeOuts***) {  
 *clearTimeout*(***targetTimeOuts***[elem.**id**]);  
 }  
 ***targetTimeOuts***[elem.**id**] = *setTimeout*(*targetMover*, ***timer***, elem);  
}  
  
*/\*------------------ NICE ANIMATIONS ------------------\*/***function** *animateAndRemove*(elem, className) {  
 elem.**classList**.add(className);  
 elem.addEventListener(**'transitionend'**, **function** *templeRemover*() {  
 **this**.removeEventListener(**'transitionend'**, *templeRemover*);  
 elem.**parentElement**.removeChild(elem);  
 });  
}  
  
**function** *addNewPopUpScore*(data, style, x, y) {  
 **let** newElem = ***document***.createElement(**'text'**);  
 newElem.**innerHTML** = data;  
 newElem.**classList**.add(**'popUpScore'**);  
 newElem.**style**.**left** = x + **'px'**;  
 newElem.**style**.**top** = y + **'px'**;  
 newElem.**style**.**fontSize** = style[**'fontSize'**] + **'vh'**;  
 newElem.**style**.**fontWeight** = style[**'fontWeight'**];  
 newElem.**style**.**color** = style[**'color'**];  
  
 ***document***.**body**.append(newElem);  
 ***window***.setTimeout(*animateAndRemove*.bind(**this**, newElem, **'fade-out'**), 100);  
}  
  
**function** *addMissclick*(x, y) {  
 **let** newElem = ***document***.createElement(**'div'**);  
 newElem.**classList**.add(**'missclickPoint'**);  
 newElem.**style**.**left** = x + **'px'**;  
 newElem.**style**.**top** = y + **'px'**;  
 ***document***.**body**.append(newElem);  
 ***console***.log(**"its OK"**);  
 ***window***.setTimeout(*animateAndRemove*.bind(**this**, newElem, **'disappear'**), 10);  
}  
  
**function** *trigger*(elem, color = ***DEFAULT\_TEXT\_COLOR***) {  
 elem.**style**.**transform** = **'scale('** + 1.2.toString() + **')'**;  
 elem.**style**.**color** = color;  
 elem.addEventListener(**'transitionend'**, **function** () {  
 elem.**style**.**transform** = **'scale(1)'**;  
 elem.**style**.**color** = ***DEFAULT\_TEXT\_COLOR***;  
 }  
 );  
}  
  
**function** *mouseTrigger*(value, deg) {  
 ***aimElem***.**style**.**transform** = **'scale('** + value + **')'** + **' '** + **'rotate('** + deg + **')'**;  
 ***aimElem***.addEventListener(**'transitionend'**, **function** () {  
 ***aimElem***.**style**.**transform** = **'scale(1)'**;  
 }  
 );  
}  
  
**function** *targetTrigger*(elem) {  
 elem.**style**.**transform** = **'scale('** + 1.2.toString() + **')'**;  
 elem.**style**.**opacity** = **'0.1'**;  
 elem.addEventListener(**'transitionend'**, **function** () {  
 elem.**style**.**transform** = **'scale(1)'**;  
 elem.**style**.**opacity** = **'1'**;  
 }  
 );  
}

Файл «updaters.js».

*/\*------------------ Varibles ------------------\*/***let *scoreLabel*** = ***document***.getElementById(**'scoreCounterLabel'**);  
**let *levelLabel*** = ***document***.getElementById(**'levelCounterLabel'**);  
  
*/\*------------------ GAME LOGIC ------------------\*/***function** *scoreUpdater*(value) {  
 **if** (value > 0) {  
 *trigger*(***scoreLabel***, ***triggerColor***[0]);  
 } **else** {  
 *trigger*(***scoreLabel***, ***triggerColor***[1]);  
 }  
 ***scoreCounter*** += value;  
 ***scoreLabel***.**innerHTML** = **'Score: '** + **scoreCounter**.toString();  
 *levelUpdater*();  
}  
  
**function** *levelUpdater*() {  
 **if** (***currentLevel*** \* ***levelUpdateValue*** \* 1000 <= ***scoreCounter***) {  
 *trigger*(***levelLabel***, ***triggerColor***[0]);  
 ***currentLevel*** += 1;  
 ***timer*** = ***Math***.max(***timer*** - ***timerStep***, ***maxSpeedTimerValue***);  
  
 } **else if** ((***currentLevel*** - 1) \* ***levelUpdateValue*** \* 1000 > ***scoreCounter***) {  
 **if** (***currentLevel*** !== 1) {  
 *trigger*(***levelLabel***, ***triggerColor***[1]);  
 ***currentLevel*** -= 1;  
 ***timer*** = ***Math***.min(***timer*** + ***timerStep***, ***minSpeedTimerValue***);  
 }  
 }  
 ***levelLabel***.**innerHTML** = **'Lvl: '** + ***currentLevel***.toString();  
}  
  
**function** *misclicksUpdater*() {  
 ***misclicksCounter*** += 1;  
 ***document***.getElementById(**'misclickStatLabel'**).**innerHTML** = **`Misclicks:** ${**misclicksCounter**}**`**;  
}  
  
**function** *hitsUpdater*() {  
 ***hitsCounter*** += 1;  
 ***document***.getElementById(**'hitStatLabel'**).**innerHTML** = **`Hits:** ${**hitsCounter**}**`**;  
}  
  
**function** *targetUpdater*(value) {  
 ***targetCounter*** += value;  
 ***document***.getElementById(**'targetCounterLabel'**).**innerHTML** = **`Targets:** ${**targetCounter**}**`**;  
}  
  
  
*/\*------------------ BUTTON HANDLERS ------------------\*/***function** *addTarget*() {  
 **let** newElem = ***document***.createElement(**'div'**);  
 newElem.**classList**.add(**'target-wrapper'**);  
 newElem.**id** = **`trgN**${**newTargetId**}**`**;  
 ***newTargetId*** += 1;  
 newElem.**innerHTML** =  
 **' <div class="target trg1">\n'** +  
 **' <div class="target trg2">\n'** +  
 **' <div class="target trg3">\n'** +  
 **' <div class="target trg4">\n'** +  
 **' <div class="target trg5">\n'** +  
 **' </div>\n'** +  
 **' </div>\n'** +  
 **' </div>\n'** +  
 **' </div>\n'** +  
 **' </div>\n'** +  
 **'</div>'  
 *document***.getElementById(**'mainArea'**).appendChild(newElem);  
 **for** (**let** i = 1; i < 6; i++) {  
 newElem.querySelector(**`.trg**${i}**`**).addEventListener(**'click'**, event => *targetClickHandler*(event, i - 1));  
 }  
 *targetMover*(newElem);  
 *targetUpdater*(1);  
}  
  
**function** *deleteTarget*() {  
 **if** (***targetCounter*** > 1) {  
 **let** elem = ***document***.querySelector(**'.trg1'**);  
 elem.removeEventListener(**'click'**, *targetClickHandler*);  
 elem.remove();  
 *targetUpdater*(-1);  
 } **else** {  
 *alert*(**'Число мишеней не может быть меньше'**);  
 }  
}

Файл «handlers.js».

*/\*------------------ Varibles ------------------\*/***const *aimElem*** = ***document***.getElementById(**'aimWrapper'**);  
**let *leftAimPos*** = ***aimElem***.getBoundingClientRect().**left**;  
**let *topAimPos*** = ***aimElem***.getBoundingClientRect().**top**;  
  
*/\*------------------ Click/Keydown Handlers ------------------\*/****window***.addEventListener(**'load'**, () => {  
 ***window***.addEventListener(**'keydown'**, event => {  
 **switch** (event.**key**) {  
 **case 'ArrowLeft'**:  
 **case 'a'**:  
 **case 'ф'**:  
 ***leftAimPos*** -= ***stepAimMover***;  
 **if** (***leftAimPos*** <= 0) {  
 ***leftAimPos*** = ***screenWidth*** - ***aimElem***.**clientWidth**;  
 }  
 **break  
  
 case 'ArrowRight'**:  
 **case 'd'**:  
 **case 'в'**:  
 ***leftAimPos*** += ***stepAimMover***;  
 **if** (***leftAimPos*** >= ***screenWidth*** - ***aimElem***.**clientWidth** / 2) {  
 ***leftAimPos*** = 0;  
 }  
 **break  
  
 case 'ArrowDown'**:  
 **case 's'**:  
 **case 'ы'**:  
 ***topAimPos*** += ***stepAimMover***;  
 **if** (***topAimPos*** >= ***screenHeight*** - ***aimElem***.**clientHeight** / 2) {  
 ***topAimPos*** = 0;  
 }  
 **break  
  
 case 'ArrowUp'**:  
 **case 'w'**:  
 **case 'ц'**:  
 ***topAimPos*** -= ***stepAimMover***;  
 **if** (***topAimPos*** <= 0) {  
 ***topAimPos*** = ***screenHeight*** - ***aimElem***.**clientHeight**;  
 }  
 **break  
  
 case ' '**:  
 **const** x = ***aimElem***.getBoundingClientRect().**x** + ***aimElem***.**clientWidth** / 2; *// получаем координату X мыши* **const** y = ***aimElem***.getBoundingClientRect().**y** + ***aimElem***.**clientHeight** / 2; *// получаем координату Y мыши;* **const** mouseClickEvent = **new** *MouseEvent*(**'click'**, {  
 **'view'**: ***window***,  
 **'bubbles'**: **true**,  
 **'cancelable'**: **true**,  
 **'clientX'**: x,  
 **'clientY'**: y  
 });  
 ***document***.elementFromPoint(x, y).dispatchEvent(mouseClickEvent);  
 **break** }  
 ***aimElem***.**style**.**left** = ***leftAimPos*** + **'px'**;  
 ***aimElem***.**style**.**top** = ***topAimPos*** + **'px'**;  
 });  
});  
  
**function** *targetClickHandler*(event, index) {  
 **const** x = event.**clientX**;  
 **const** y = event.**clientY**;  
 event.stopPropagation();  
  
 **if** (index === 5) {  
 *addMissclick*(event.**clientX**, event.**clientY**);  
 *misclicksUpdater*();  
 *mouseTrigger*(**'0.8'**, **'-90deg'**);  
 } **else** {  
 **let** elem = event.**target**;  
 **while** (!(elem.**classList**.contains(**'target-wrapper'**))) {  
  
 elem = elem.**parentElement**;  
 }  
 *mouseTrigger*(**'1.4'**, **'90deg'**);  
 *targetMover*(elem);  
 *targetTrigger*(elem);  
 *hitsUpdater*();  
 }  
 **let** entry = ***Object***.keys(***scoreData***)[index];  
 *addNewPopUpScore*(entry, ***scoreData***[entry], x, y);  
 *scoreUpdater*(*parseInt*(entry));  
}  
  
***document***.getElementById(**'mainArea'**).addEventListener(**'click'**,  
 event => *targetClickHandler*(event, 5));  
  
***window***.addEventListener(**'mousemove'**, **function** (event) {  
 ***document***.getElementById(**'aimWrapper'**).**style**.**left** = event.**clientX** - 0.03 \* ***Math***.min(***screenHeight***, ***screenWidth***) + **'px'**;  
 ***document***.getElementById(**'aimWrapper'**).**style**.**top** = event.**clientY** - 0.03 \* ***Math***.min(***screenHeight***, ***screenWidth***) + **'px'**;  
})

Файл «styles.css».

*/\*------------------ Common ------------------\*/*:**root** {  
 **--minMeas**: **min**(1**vh**, 1**vw**);  
}  
  
::**-webkit-scrollbar** {  
 **width**: 0;  
 **height**: 0;  
}  
  
**body** {  
 **display**: **flex**;  
 **flex-direction**: **column**;  
 **align-items**: **center**;  
 **justify-content**: **center**;  
  
 **width**: 100**vw**;  
 **height**: 100**vh**;  
 **margin**: 0;  
  
 **font-family**: **'Verdana'**, **serif**;  
 **font-size**: **calc**(**var**(**--minMeas**) \* 2);  
 **-webkit-user-select**: **none**;  
 **-khtml-user-select**: **none**;  
 **-moz-user-select**: **none**;  
 **-ms-user-select**: **none**;  
 **-o-user-select**: **none**;  
 **user-select**: **none**;  
  
 **overflow**: **hidden**;  
 **-ms-overflow-style**: **none**;  
 **scrollbar-width**: **none**;  
 **cursor**: **none**;  
}  
  
**h1** {  
 **-webkit-animation**: **blink-header** 1.6**s linear infinite**;  
 **animation**: **blink-header** 1.6**s linear infinite**;  
 **align-items**: **center**;  
 **justify-content**: **center**;  
 **display**: **flex**;  
 **margin**: 0;  
  
 **font-weight**: **bold**;  
 **font-size**: **calc**(**var**(**--minMeas**) \* 6);  
}  
  
  
**@-webkit-keyframes blink-header** {  
 **from** {  
 **scale**: 1;  
 }  
  
 **50%** {  
 **scale**: 1.3;  
 }  
  
 **to** {  
 **scale**: 1;  
 }  
}  
  
**@keyframes blink-header** {  
 **from** {  
 **scale**: 1;  
 }  
  
 **50%** {  
 **scale**: 1.3;  
 }  
  
 **to** {  
 **scale**: 1;  
 }  
}  
  
  
**h2** {  
 **font-weight**: **bold**;  
 **margin**: 0;  
  
 **font-size**: **calc**(**var**(**--minMeas**) \* 3.5);  
}  
  
**h3** {  
 **font-weight**: **bold**;  
 **margin**: 0;  
  
 **font-size**: **calc**(**var**(**--minMeas**) \* 2.5);  
}  
  
**h4** {  
 **font-weight**: **normal**;  
 **font-size**: **calc**(**var**(**--minMeas**) \* 2.3);  
 **margin**: 0;  
}  
  
**button** {  
 **font-size**: **calc**(**var**(**--minMeas**) \* 1.5);  
 **font-weight**: **bolder**;  
 **background-color**: **transparent**;  
 **border**: **none**;  
  
 **width**: **calc**(**var**(**--minMeas**) \* 3);  
 **height**: **calc**(**var**(**--minMeas**) \* 3);  
 **text-align**: **center**;  
  
}  
  
*/\*------------------ Header ------------------\*/***#header** {  
 **display**: **grid**;  
 **grid-template-columns**: 1**fr** 1**fr** 1**fr**;  
  
 **justify-content**: **space-around**;  
 **justify-items**: **center**;  
 **align-content**: **center**;  
 **align-items**: **center**;  
  
 **width**: 100%;  
 **padding-top**: 2**vh**;  
 **height**: **fit-content**;  
  
 **z-index**: 9999;  
}  
  
.**statData** {  
 **display**: **flex**;  
 **flex-direction**: **column**;  
 **flex-wrap**: **nowrap**;  
 **align-items**: **flex-start**;  
 **justify-content**: **flex-start**;  
  
 **font-size**: **calc**(**var**(**--minMeas**) \* 1.4);  
 **font-weight**: **normal**;  
 **position**: **relative**;  
 **height**: **fit-content**;  
 **width**: **fit-content**;  
}  
  
**#statTargetsBlock** {  
 **gap**: **calc**(**var**(**--minMeas**) \* 2);  
 **display**: **flex**;  
 **flex-direction**: **row**;  
 **justify-content**: **space-between**;  
 **align-items**: **center**;  
}  
  
**#scoreCounterLabel**, **#levelCounterLabel** {  
 **transition**: **transform** 0.3**s**, **color** 0.2**s**;  
}  
  
*/\*------------------ BodyBlock ------------------\*/***#mainArea** {  
 **display**: **block**;  
 **width**: 100%;  
 **height**: 100%;  
}  
  
*/\*------------------ AnimatedBlocks ------------------\*/***#scoreViewer** {  
 **position**: **absolute**;  
 **z-index**: 99999;  
}  
  
*/\*------------------ TargetBlock ------------------\*/*.**target-wrapper** {  
 **display**: **block**;  
 **position**: **absolute**;  
 **width**: **fit-content**;  
 **height**: **fit-content**;  
  
 **left**: **calc**(100**vw** / 2);  
 **top**: **calc**(100**vh** / 2);  
 **background-color**: **black**;  
 **z-index**: 100;  
 **transition**: **left** 0.5**s ease-in-out**, **top** 0.5**s ease-in-out**, **transform** 2**s**, **opacity** 1**s**;  
}  
  
.**target** {  
 **display**: **flex**;  
 **position**: **absolute**;  
 **border-radius**: 50%;  
  
 **flex-direction**: **column**;  
 **justify-content**: **center**;  
 **align-content**: **center**;  
 **align-items**: **center**;  
}  
  
.**trg1** {  
 **position**: **absolute**;  
  
 **height**: **min**(13**vh**, 13**vw**);  
 **width**: **min**(13**vh**, 13**vw**);  
  
 **background-color**: **white**;  
 **-webkit-box-shadow**: 5**px** 10**px** 60**px** 0 **rgba**(34, 60, 80, 0.4);  
 **-moz-box-shadow**: 5**px** 10**px** 60**px** 0 **rgba**(34, 60, 80, 0.4);  
 **box-shadow**: 5**px** 10**px** 60**px** 0 **rgba**(34, 60, 80, 0.4);  
}  
  
.**trg2** {  
 **background-color**: **black**;  
 **height**: **min**(10**vh**, 10**vw**);  
 **width**: **min**(10**vh**, 10**vw**);  
 **border**: **none**;  
 **z-index**: 1;  
}  
  
.**trg3** {  
 **background-color**: **#2219B2**;  
 **height**: **min**(7**vh**, 7**vw**);  
 **width**: **min**(7**vh**, 7**vw**);  
 **border**: **none**;  
 **z-index**: 3;  
}  
  
.**trg4** {  
 **background-color**: **#EA0037**;  
 **height**: **min**(4**vh**, 4**vw**);  
 **width**: **min**(4**vh**, 4**vw**);  
 **border**: **none**;  
 **z-index**: 2;  
}  
  
.**trg5** {  
 **background-color**: **yellow**;  
 **height**: **min**(1.7**vh**, 1.7**vw**);  
 **width**: **min**(1.7**vh**, 1.7**vw**);  
 **border**: **none**;  
 **z-index**: 10;  
}  
  
  
*/\*------------------ AimBlock ------------------\*/***#aimImg** {  
 **width**: **min**(10**vw**, 10**vh**);  
 **height**: **min**(10**vw**, 10**vh**);  
 **position**: **relative**;  
}  
  
**#aimWrapper** {  
 **position**: **absolute**;  
 **animation**: **transition**;  
 **z-index**: 9999;  
}  
  
.**popUpScore** {  
 **display**: **block**;  
 **position**: **absolute**;  
 **opacity**: 1;  
 **transition**: **opacity** 1**s**, **transform** 1**s**;  
 **z-index**: 9999;  
}  
  
.**popUpScore**.**fade-out** {  
 **opacity**: 0;  
 **transform**: **translate**(0, -20**vh**);  
}  
  
*/\*------------------ Footer ------------------\*/***footer** {  
 **display**: **grid**;  
 **grid-template-columns**: 1**fr** 2**fr** 1**fr**;  
  
 **justify-items**: **center**;  
 **align-content**: **center**;  
 **align-items**: **center**;  
  
 **width**: 100%;  
 **padding-bottom**: 3**vh**;  
}  
  
**#statDataBlock** {  
 **justify-self**: **left**;  
 **padding-left**: **var**(**--minMeas**);  
}  
  
**#creator** {  
 **justify-self**: **right**;  
 **align-items**: **flex-end**;  
 **padding-right**: **var**(**--minMeas**);  
}  
  
.**targetBtns**, **#aim** {  
 **width**: **calc**(**var**(**--minMeas**) \* 6);  
 **height**: **calc**(**var**(**--minMeas**) \* 6);  
 **cursor**: **pointer**;  
 **display**: **block**;  
 **background-color**: **transparent**;  
 **position**: **relative**;  
 **transition**: **transform** 0.8**s**;  
 **border-radius**: 50%;  
}  
  
.**targetBtns**:**hover** {  
 **transform**: **rotate**(180**deg**);  
}  
  
.**targetBtns**::**before**,  
.**targetBtns**::**after**, **#aim**::**before**, **#aim**::**after** {  
 **content**: **''**;  
 **display**: **block**;  
 **position**: **absolute**;  
 **width**: **calc**(**var**(**--minMeas**) \* 3);  
 **height**: **calc**(**var**(**--minMeas**) \* 0.6);  
 **border-radius**: 7%;  
 **background**: **black**;  
 **vertical-align**: **middle**;  
 **justify-self**: **center**;  
 **top**: 0;  
 **left**: 0;  
 **right**: 0;  
 **bottom**: 0;  
 **margin**: **auto**;  
 **transition**: **background** 0.5**s**;  
}  
  
  
.**targetBtns**::**before**, **#aim**::**before** {  
 **transform**: **rotate**(90**deg**);  
}  
  
**#delTargetBtn**.**targetBtns**::**before** {  
 **display**: **none**;  
}  
  
**#addTargetBtn**.**targetBtns**::**before** {  
 **transform**: **rotate**(90**deg**);  
}  
  
**#aim** {  
 **background-color**: **transparent**;  
 **margin**: **auto**;  
}  
  
**#aimWrapper** {  
 **display**: **block**;  
 **position**: **absolute**;  
 **width**: **fit-content**;  
 **height**: **fit-content**;  
 **border**: **black calc**(**var**(**--minMeas**) \* 0.6) **solid**;  
 **justify-content**: **center**;  
 **align-content**: **center**;  
 **background-color**: **transparent**;  
 **top**: 0;  
 **right**: 0;  
 **bottom**: 0;  
 **left**: 0;  
 **z-index**: 9999;  
 **pointer-events**: **none**;  
 **transition**: **transform** 0.3**s**, **background-color** 0.3**s**;  
}  
  
.**missclickPoint**{  
 **display**: **block**;  
 **position**: **absolute**;  
 **width**: **calc**(**var**(**--minMeas**) \* 5);  
 **height**: **calc**(**var**(**--minMeas**) \* 5);  
 **background-color**: **#EA0037**;  
 **border-radius**: 50%;  
 */\*filter: blur(5px);  
  
 \*/* **opacity**: 1;  
 **transition**: **opacity** 15**s**;  
 **z-index**: 0;  
}  
  
.**missclickPoint**.**disappear**{  
 **opacity**: 0;  
}