

Разработка полнофункциональной веб-игры «Танковая дуэль» с использованием HTML, CSS и JavaScript

Языков Кирилл Александрович

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

Целью исследования является разработка полнофункциональной двухигровой аркады «Танковая дуэль». В проекте использованы нативные веб-технологии — HTML5, CSS3 и JavaScript, а также всё графическое и звуковое сопровождение, кроме фоновой музыки, генерируется программно без внешних зависимостей. Разработанное приложение демонстрирует высокую автономность, читаемость кода и образовательную ценность, позволяя использовать его как учебный пример или готовый продукт для локального соревнования.

Ключевые слова: HTML, CSS, JavaScript, веб-приложение, аркада, танки, Web Audio API, игра.

Development of a full-featured web game «Tank Duel» using HTML, CSS and JavaScript

Yazykov Kirill Alexandrovich

Sholom-Aleichem Priamursky State University

Student

Abstract

The purpose of the research is to develop a fully functional two-game arcade game "Tank Duel". The project uses native web technologies - HTML5, CSS3 and JavaScript, as well as all graphics and sound, except for background music, generated programmatically without external dependencies. The developed application demonstrates high autonomy, code readability and educational value, allowing it to be used as a training example or a ready-made product for a local competition.

Keywords: HTML, CSS, JavaScript, Web application, arcade, tanks, Web Audio API, game.

1 Введение

1.1 Актуальность

Интерактивные веб-игры, реализованные без внешних зависимостей и использующие исключительно нативные веб-технологии, получают всё большее распространение. Такие проекты служат не только средством развлечения, но и эффективным инструментом обучения основам

программирования, особенно в области клиентской веб-разработки. «Танковая дуэль» представляет собой классическую аркаду с простой, но глубокой механикой, ориентированную на локальное соревнование двух игроков. Возможность управлять танками с одной клавиатуры делает игру доступной в условиях ограниченных ресурсов, а использование визуальных и звуковых эффектов без внешних файлов подчёркивает эффективность современных возможностей браузера.

1.2 Обзор исследований

И. В. Смирнов в своей статье рассмотрел применение HTML5 и CSS3 для создания анимированных веб-интерфейсов, подчеркнув важность использования ключевых кадров для реализации сложных анимаций [1]. А. С. Петрова исследовала методы перетаскивания и обработки указателей в JavaScript, что положено в основу интерактивности многих игр [2]. Д. А. Кузнецов представил программную реализацию системы управления мультимедийным контентом в браузере, включая воспроизведение аудио и управление громкостью с помощью элементов управления, что является важным аспектом для создания атмосферных приложений [3]. Е. М. Григорьева и М. П. Лебедев в своей работе описали подходы к созданию адаптивных веб-приложений с использованием Flexbox и Grid, включая панели управления и игровые поля [4]. П. Н. Тихонов подчеркнул важность оптимизации анимаций с помощью requestAnimationFrame для обеспечения плавности даже при большом количестве объектов [5].

1.3 Цель исследования

Целью исследования является разработка полнофункциональной двухигровой аркады «Танковая дуэль» с использованием HTML, CSS и JavaScript.

2 Материалы и методы

Для разработки приложения использовалась интегрированная среда Visual Studio Code, поддерживающая подсветку синтаксиса и отладку для веб-технологий. Архитектура приложения реализована в соответствии с принципом разделения ответственности:

- HTML5: структура экранов: главное меню, игровой экран, финальный экран, модальное окно с правилами.
- CSS3: стилизация интерфейса, анимации, градиентные фоны, адаптивность.
- JavaScript: игровая логика, обработка событий клавиатуры, коллизии, анимации, звук, работа с canvas.

Все игровые элементы (танки, пули, взрывы) отрисовываются на HTML5 <canvas>. Звуковые эффекты создаются через Web Audio API — встроенный браузерный инструмент для синтеза звука [6]. Единственное

обращение к внешнему файлу — попытка загрузки background.mp3 для фоновой музыки, которая не влияет на геймплей и может быть отключена [7].

3 Результаты

Разработанное интерактивное веб-приложение «Танковая дуэль» представляет собой полноценную браузерную аркаду, реализованную исключительно на нативных веб-технологиях (HTML5, CSS3, JavaScript) без использования сторонних библиотек. Приложение функционирует автономно: после загрузки HTML-файла не требует подключения к интернету и корректно отображается на устройствах с различными разрешениями экрана благодаря центрированной адаптивной компоновке на основе Flexbox.

3.1 Архитектура приложения

Приложение состоит из трёх логически разделённых экранов, каждый из которых реализован как абсолютный позиционированный блок с центрированным содержимым.

Главное меню (рис. 1) содержит заголовок с эмодзи «🏆 Танковая дуэль 🏆», краткое описание игровой механики и две интерактивные кнопки: «Начать игру» и «📖 Правила игры». При нажатии на последнюю открывается модальное окно с развёрнутыми правилами, оформленное в едином стиле с градиентным фоном и полупрозрачной заливкой. Дизайн интерфейса реализован с использованием шрифтовых стеков Orbitron (для заголовков) и Echo 2 (для основного текста), что придаёт приложению футуристическую, «игровую» эстетику.

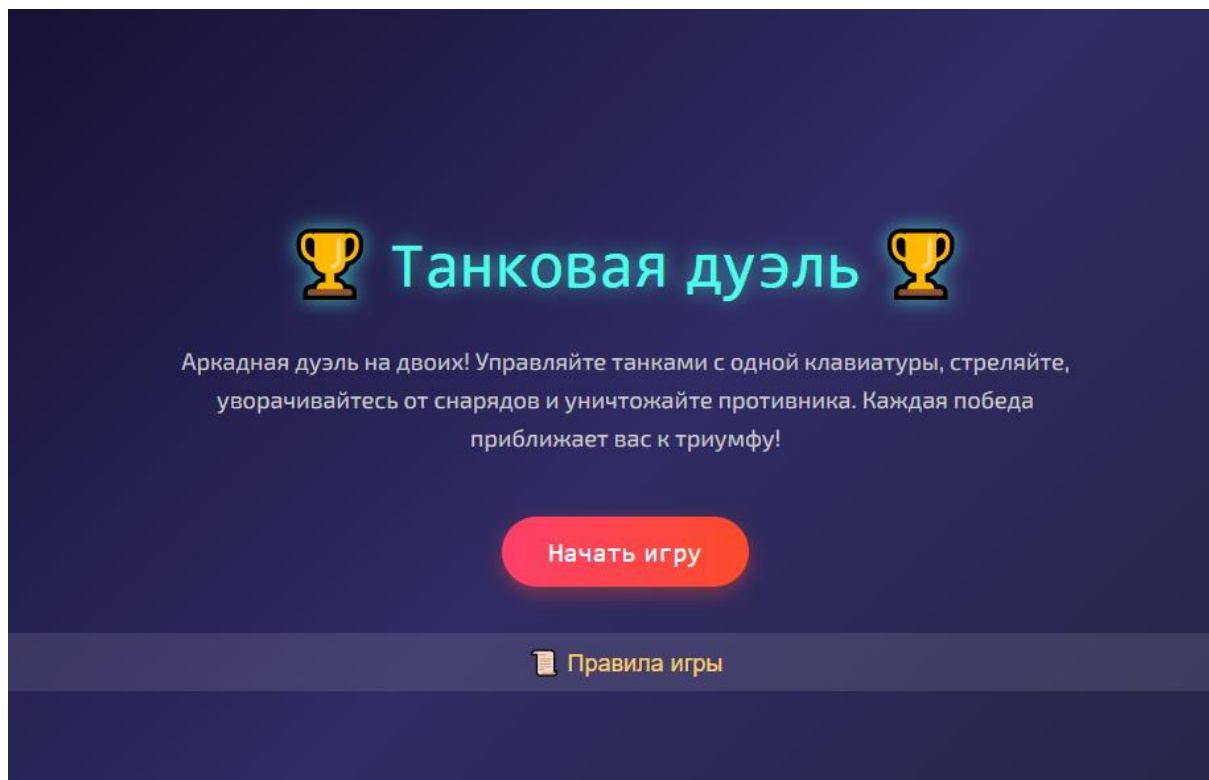


Рисунок 1 – Главное меню игры

Игровой экран (рис. 2) состоит из HTML5-элемента `<canvas>` размером 800×600 пикселей, на котором программно рисуется всё содержимое: фон, препятствия, зоны спавна, танки, снаряды и визуальные эффекты. По бокам от холста расположены статические панели с информацией об игроках: схема управления клавишами, индикаторы жизней (в виде эмодзи ❤️), счёт и визуальный индикатор перезарядки — полоса, ширина которой динамически изменяется в зависимости от времени, прошедшего с последнего выстрела. В верхней части экрана отображается таймер, отсчитывающий 3 минуты игрового времени.

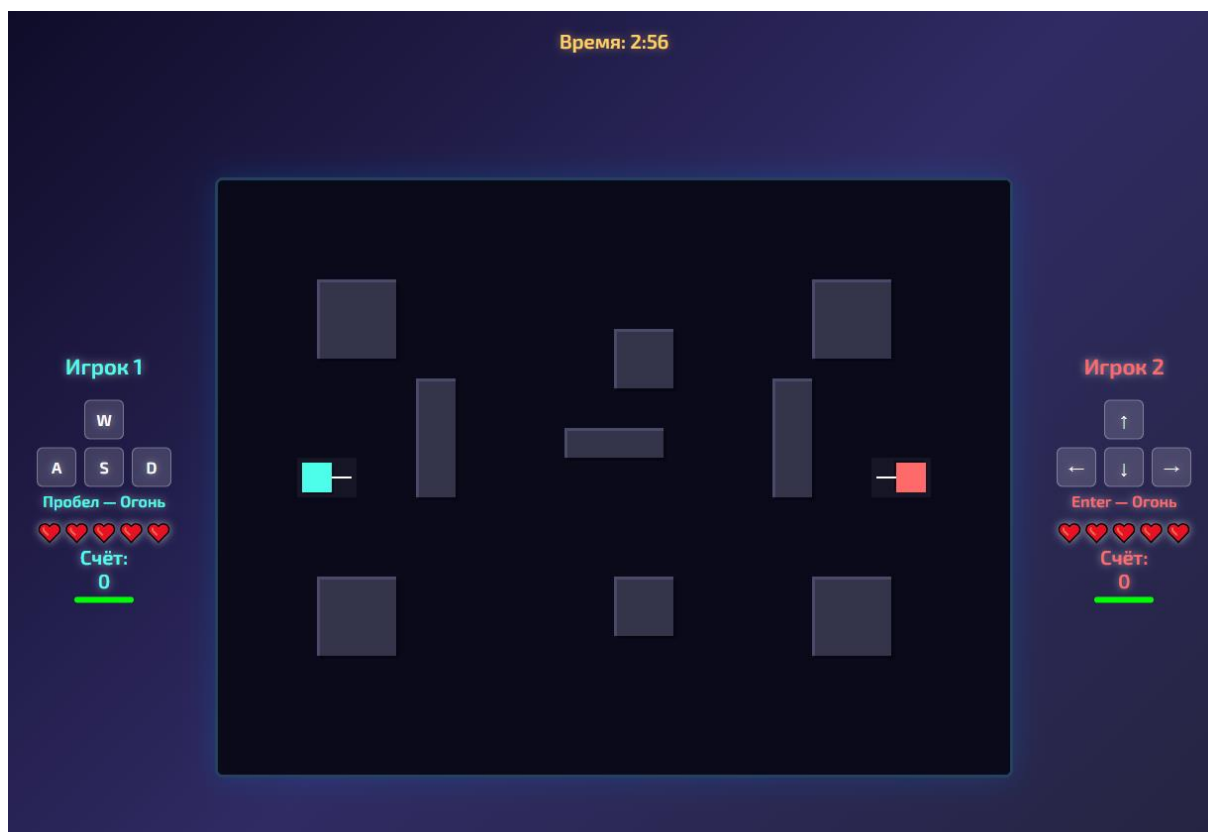


Рисунок 2 – Игровой экран игры

Финальный экран (рис. 3) активируется по завершении игры и отображает имя победителя («Победил Игрок 1!» или «Победил Игрок 2!») или же если игроки набрали одинаковое количество очков по истечению времени, то отображается надпись: «Ничья» с применением градиентного текста и анимации пульсации (`@keyframes pulse`). Также присутствует кнопка возврата в главное меню, ведущая к полному сбросу игрового состояния.

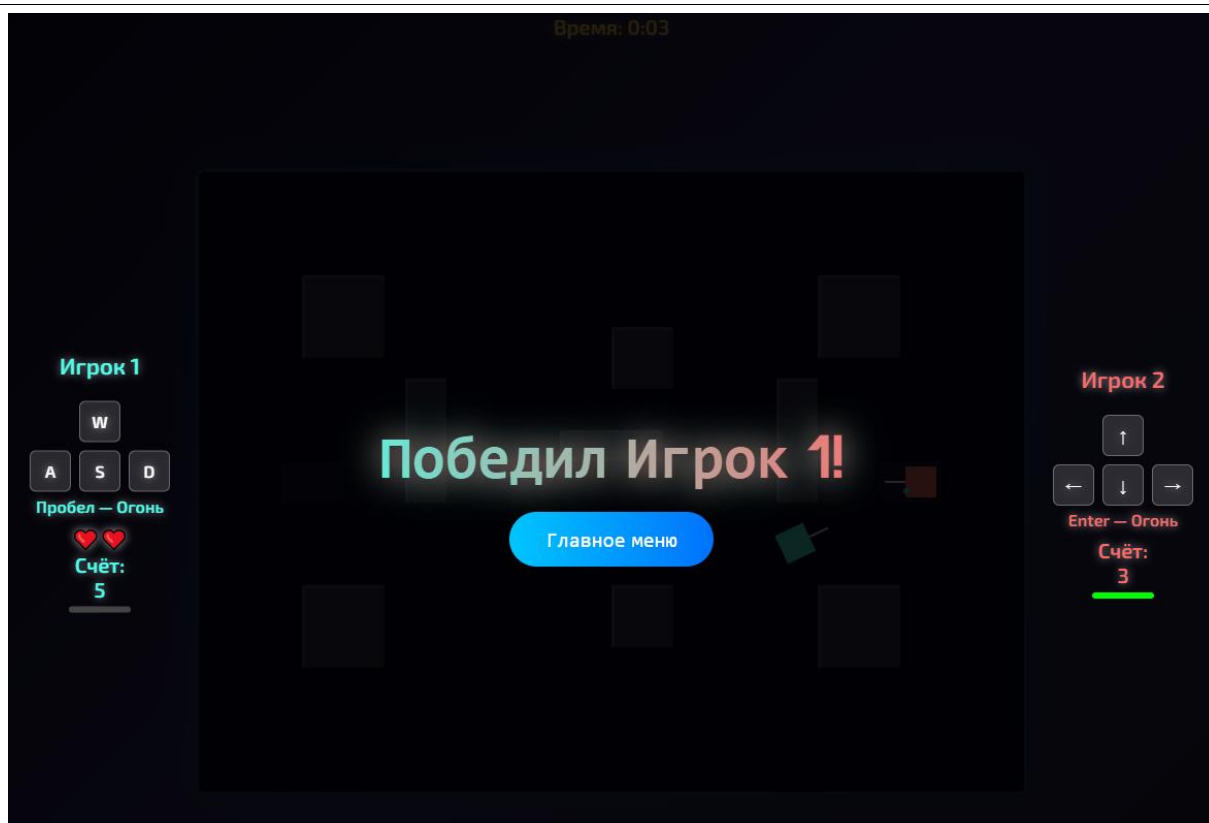


Рисунок 3 – Финальный экран игры

3.2 Правила игры и система начисления очков

Игра рассчитана на двух игроков, использующих одну клавиатуру (рис. 4). Управление разделено следующим образом:

- Игрок 1: движение — клавиши W, A, S, D, стрельба — Пробел;
- Игрок 2: движение — стрелки (↑, ←, ↓, →), стрельба — Enter.

Каждый игрок располагает пятью единицами здоровья, визуализированными как пять сердец (♥). При попадании снаряда в танк:

- поражённый игрок теряет одну жизнь;
- атакующий получает одно очко;
- уничтоженный танк мгновенно исчезает и возрождается в своей зоне спавна через 1,5 секунды с эффектом мерцания.

Победа присуждается игроку, первым набравшему 5 очков. Если по истечении трёх минут счёт не достиг этого значения, побеждает игрок с большим количеством очков (в случае равенства — объявляется ничья).

Зоны спавна — прямоугольные области в левой (для Игрока 1) и правой (для Игрока 2) частях поля — недоступны для вторжения. При попытке входа противником его позиция корректируется алгоритмом отталкивания, что реализует механику «защиты базы».

Препятствия (всего 9) — непроходимые блоки, размещённые симметрично на поле. Снаряды, попадающие в них, уничтожаются с визуальным эффектом взрыва, но не наносят урона игрокам.

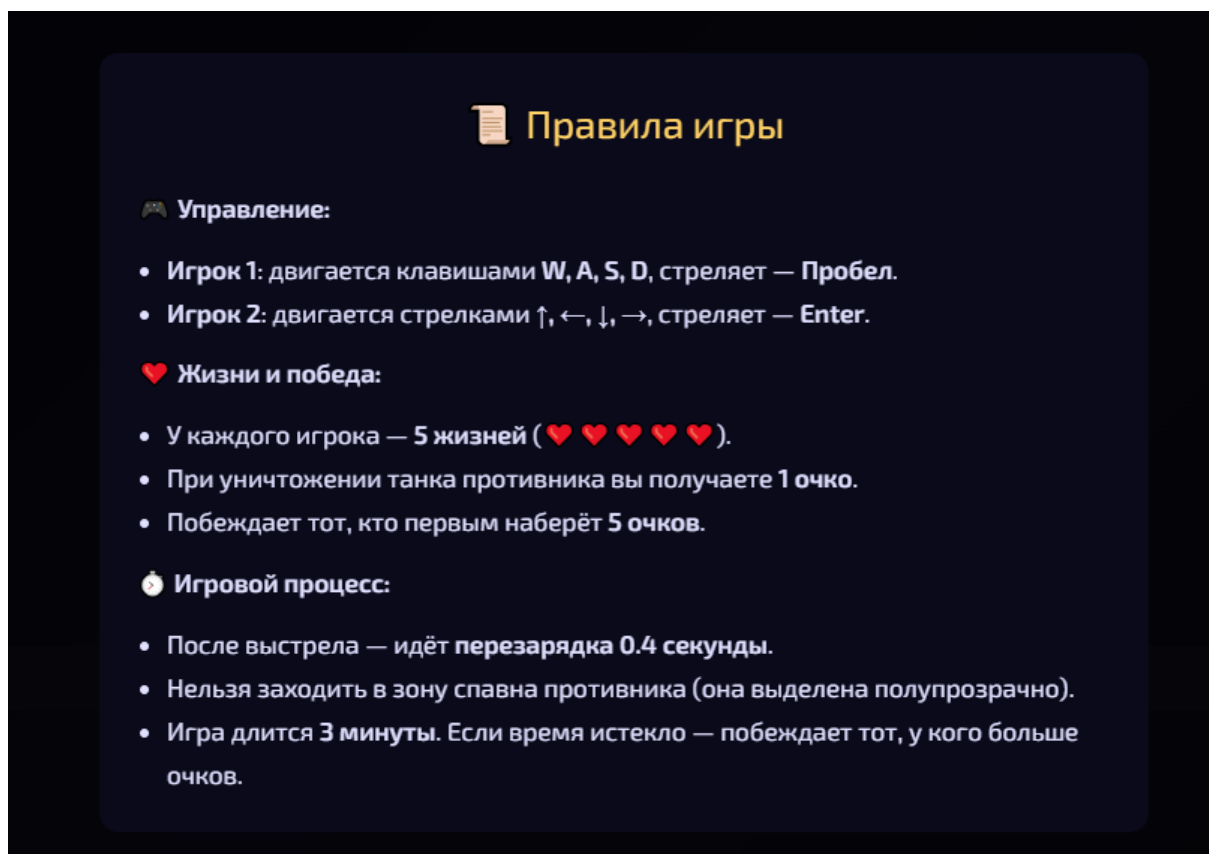


Рисунок 4 – Правила игры

3.3 Основные функции и их реализация

Поддержка русской и английской раскладок:

Система ввода реализована через глобальный объект `keyState`, который отслеживает нажатие клавиш. Для поддержки обеих раскладок используется маппинг: символы «ц, ф, ы, в» трактуются как «W, A, S, D» соответственно. Это обеспечивает полную совместимость с русскоязычными пользователями без дублирования логики:

```
const keyState = {};
const keyMap = {
  'w': 'w', 'ц': 'w',
  'a': 'a', 'ф': 'a',
  's': 's', 'ы': 's',
  'd': 'd', 'в': 'd',
  ' ': 'space',
  'arrowup': 'arrowup',
  'arrowdown': 'arrowdown',
  'arrowleft': 'arrowleft',
  'arrowright': 'arrowright',
  'enter': 'enter'};
window.addEventListener('keydown', (e) => {
  const keyName = e.key.toLowerCase();
  const mappedKey = keyMap[keyName] || keyName;
  keyState[mappedKey] = true;
  if (!gameStarted) {
    gameStarted = true;
    startTimer();
  });});
```

Функция `playTone()` — генерация звука через Web Audio API.

Эта функция реализует встроенные звуки Windows программно, без .wav- или .mp3-файлов. Web Audio API — это встроенный в браузер интерфейс для создания и обработки аудиосигналов в реальном времени:

```
function playTone(f, d, t = 'square', g = 0.2) {
    if (!window.AudioContext) return;
    const audioCtx = new AudioContext();
    const o = audioCtx.createOscillator();
    const gn = audioCtx.createGain();
    o.type = t;
    o.frequency.value = f;
    gn.gain.value = g;
    gn.gain.exponentialRampToValueAtTime(0.001, audioCtx.currentTime + d);
    o.connect(gn);
    gn.connect(audioCtx.destination);
    o.start();
    o.stop(audioCtx.currentTime + d);}
```

Примеры:

- `playTone(400, 0.1, 'square')` — выстрел;
- `playTone(100, 0.3, 'sawtooth')` — взрыв;
- `playTone(200, 0.05, 'sine')` — щелчок (попытка стрельбы при зарядке).

Функция `Tank.shoot()` — стрельба с перезарядкой:

```
shoot() {
    if (Date.now() - this.lastShot < RELOAD_TIME) {
        playClick();
        return null;
    }
    const bulletX = this.x + Math.cos(this.angle) * (TANK_SIZE/2 + 20);
    const bulletY = this.y + Math.sin(this.angle) * (TANK_SIZE/2 + 20);
    this.lastShot = Date.now();
    playTone(400, 0.1, 'square', 0.15);
    return new Bullet(bulletX, bulletY, this.angle, this.owner);}
```

Обработка внешней фоновой музыки:

```
function getBgMusic() {
    if (!bgMusic) {
        bgMusic = new Audio();
        bgMusic.src = 'background.mp3';
        bgMusic.loop = true;
        bgMusic.volume = 0.3;
    }
    return bgMusic;}
```

Если файл `background.mp3` отсутствует, игра продолжает работать без фоновой музыки, но со всеми звуковыми эффектами.

Визуальные эффекты: `Explosion` и `Particle` (рис.5). При попадании создаётся объект `Explosion`, рисующий радиальный градиент (жёлтый → оранжевый → красный), и 20 частиц (`Particle`) — маленькие квадраты, разлетающиеся с затуханием:

```
class Explosion {
    constructor(x, y) {
        this.x = x; this.y = y; this.radius = 10; this.maxRadius = 50;
```

```

        this.growthRate = 2; this.alpha = 1; this.fadeRate = 0.03;
        for (let i = 0; i < 20; i++) {
            particles.push(new Particle(x, y, Math.random() > 0.5 ? '#ff6' :
'#f60'));
        }
        update() {
            this.radius += this.growthRate; this.alpha -= this.fadeRate;
            return this.alpha > 0;
        }
        draw() {
            ctx.save(); ctx.globalAlpha = this.alpha;
            ctx.beginPath(); ctx.arc(this.x, this.y, this.radius, 0, Math.PI *
2);
            const gradient = ctx.createRadialGradient(this.x, this.y, 0, this.x,
this.y, this.radius);
            gradient.addColorStop(0, 'yellow'); gradient.addColorStop(0.5,
'orange'); gradient.addColorStop(1, 'red');
            ctx.fillStyle = gradient; ctx.fill(); ctx.restore();
        }
    }
}

```

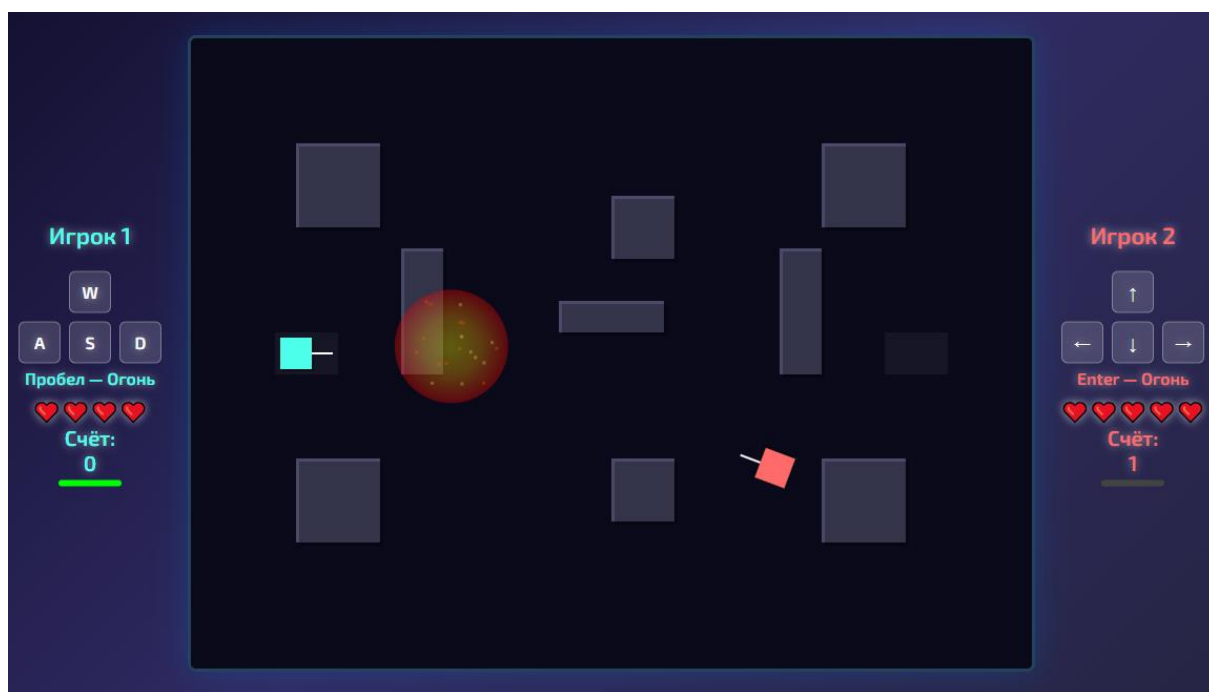


Рисунок 5 – Визуальный эффект взрыва при стрельбе

Защита зон спавна

В методе `Tank.update()` реализована проверка: если танк пытается войти в чужую зону спавна — движение блокируется, и координаты корректируются.

Функция `checkCollisions()` — обработка столкновений

Проверяет попадание снарядов в танки, препятствия и выход за пределы поля. При попадании:

- уменьшается число жизней;
- вызывается метод `hit()` у танка;
- создаётся объект `Explosion`.

```
function checkCollisions() {
  for (let i = bullets.length - 1; i >= 0; i--) {
    const bullet = bullets[i];
    if (!bullet.active) continue;
    let hit = false;
    if (tank1.alive && bullet.owner !== 'p1') {
      const dx = bullet.x - tank1.x;
      const dy = bullet.y - tank1.y;
      const dist = Math.sqrt(dx*dx + dy*dy);
      if (dist < TANK_SIZE/2 + bullet.radius) {
        hit = true;
        bullet.active = false;
        tank1.hit();
        lives1--;
        updateLives();
        if (lives1 >= 0) {
          score2++;
          updateScore();
          if (score2 >= MAX_SCORE) {
            showFinalScreen('2');}}}}
      // Аналогично для танка 2
    if (hit) {
      explosions.push(new Explosion(bullet.x, bullet.y));
      playTone(100, 0.3, 'sawtooth', 0.3);}}}
```

Функция `updateReloadBars()` — визуальный индикатор перезарядки.

Обновляет ширину и цвет полосы перезарядки над каждым танком на основе времени с последнего выстрела.:

```
function updateReloadBars() {
  const now = Date.now();
  const fill1 = Math.max(0, (now - tank1.lastShot) / RELOAD_TIME);
  const fill2 = Math.max(0, (now - tank2.lastShot) / RELOAD_TIME);
  reload1Fill.style.width = `${Math.min(100, fill1 * 100)}%`;
  reload2Fill.style.width = `${Math.min(100, fill2 * 100)}%`;
  if (fill1 >= 1) reload1Fill.style.background = '#0f0';
  else reload1Fill.style.background = '#444';
  if (fill2 >= 1) reload2Fill.style.background = '#0f0';
  else reload2Fill.style.background = '#444';}
```

Функция `startGame()` — инициализация игры.

Скрывает главное меню, сбрасывает счёт, жизни, создаёт танки и запускает игровой цикл.:

```
function startGame() {
  menuScreen.style.display = 'none';
  document.getElementById('gameContainer').style.display = 'flex';
  score1 = 0; score2 = 0; lives1 = 5; lives2 = 5;
  updateScore(); updateLives();
  tank1.respawn();
  tank2.respawn();
  gameStarted = true;
  startTimer();
  gameLoop();}
```

Функция `gameLoop()` — основной игровой цикл

Использует `requestAnimationFrame` для синхронизации с частотой обновления экрана. На каждом кадре:

- очищает холст;
- отрисовывает фон, препятствия, зоны спавна;
- обновляет положение танков и снарядов;
- проверяет столкновения;
- отрисовывает взрывы и частицы;
- обновляет индикаторы перезарядки.

```
function gameLoop() {
  if (gameEnded) return;
  ctx.clearRect(0, 0, canvas.width, canvas.height);
  ctx.fillStyle = '#0a0a1a';
  ctx.fillRect(0, 0, canvas.width, canvas.height);
  ctx.fillStyle = '#4a4a6a';
  for (let obs of obstacles) {
    ctx.fillRect(obs.x, obs.y, obs.w, obs.h);
    ctx.fillStyle = 'rgba(0,0,0,0.3)';
    ctx.fillRect(obs.x + 3, obs.y + 3, obs.w, obs.h);
    ctx.fillStyle = '#4a4a6a';
  }
  ctx.fillStyle = 'rgba(255,255,255,0.05)';
  for (let zone of spawnZones) {
    ctx.fillRect(zone.x, zone.y, zone.w, zone.h);
  }
  updateControls();
  tank1.update();
  tank2.update();
  tank1.draw();
  tank2.draw();
  bullets.forEach(b => { b.update(); b.draw(); });
  bullets = bullets.filter(b => !b.isOffscreen() && b.active);
  explosions.forEach(e => e.draw());
  explosions = explosions.filter(e => e.update());
  particles.forEach(p => p.draw());
  particles = particles.filter(p => p.update());
  checkCollisions();
  updateReloadBars();
  gameLoopId = requestAnimationFrame(gameLoop);
}
```

Функция `showFinalScreen()` — финальный экран.

Отображает победителя и проигрывает звуковой сигнал:

```
function showFinalScreen(winner) {
  gameEnded = true;
  winnerText.textContent = `Победил Игрок ${winner}!`;
  finalScreen.style.display = 'flex';
  playTone(523, 0.2, 'sine', 0.3);
  setTimeout(() => playTone(659, 0.2, 'sine', 0.3), 220);
  setTimeout(() => playTone(784, 0.3, 'sine', 0.3), 440);
  if (gameLoopId) {
    cancelAnimationFrame(gameLoopId);
    gameLoopId = null;
  }
  // Останавливаем музыку
  if (bgMusic) {
    bgMusic.pause();
  }
}
```

4 Выводы

В ходе исследования было создано полностью функциональное веб-приложение «Танковая дуэль», демонстрирующее возможности HTML5, CSS3 и JavaScript для создания автономных, интерактивных игр. Все графические и звуковые эффекты реализованы программно, без внешних изображений или аудиофайлов. Единственное внешнее обращение — загрузка background.mp3 для фоновой музыки, которая не обязательна для работы игры. Архитектура проекта отличается читаемостью, модульностью и производительностью. Приложение может использоваться как учебный пример или как готовый продукт для локального соревнования двух игроков.

Библиографический список

1. Смирнов И.В. Использование CSS3 для создания анимированных веб-интерфейсов // Веб-разработка и дизайн. 2023. № 3. С. 12-18.
2. Петрова А.С. Разработка интерактивных игр на JavaScript: методы управления событиями мыши // Программирование для начинающих. 2022. № 7. С. 34-40.
3. Кузнецов Д.А. Программная реализация системы управления мультимедийным контентом в браузере // Мультимедиа технологии. 2021. № 5. С. 56-61.
4. Григорьева Е.М., Лебедев М.П. Создание адаптивных веб-приложений с использованием Flexbox и Grid // Современные веб-технологии. 2020. № 9. С. 22-27.
5. Тихонов П.Н. Оптимизация производительности веб-анимаций // Производительность веб-приложений. 2024. № 2. С. 45-50.
6. MDN Web Docs. Web Audio API. URL: https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API (дата обращения: 04.12.2025)
7. Материалы разработки полнофункциональной веб-игры «Танковая дуэль» URL: <https://drive.google.com/drive/folders/1cPu3bVvyirRffEJW63TtkfqD33wu29u> (дата обращения: 04.12.2025)