

CENTRO UNIVERSITÁRIO FEI

Gabriel Fabiani

Yan Argolo

Vitor Souza

PROJETO VARELA'S RACE

Relatório Final do Projeto apresentado ao
Centro Universitário FEI, como parte dos
requisitos da Disciplina “CCP120”

São Bernado do Campo

2025

SUMÁRIO

1	INTRODUÇÃO	3
2	DESENVOLVIMENTO	3
3	RESULTADOS.....	5
4	CONCLUSÕES	6
5	REFERÊNCIAS	7

1 INTRODUÇÃO

O projeto consiste em um jogo de corrida desenvolvido para navegador, utilizando HTML, CSS e JavaScript. O objetivo foi criar uma aplicação interativa que simula uma corrida contra o tempo, onde o jogador deve desviar de carros inimigos e alcançar a linha de chegada antes que o cronômetro termine.

Tecnologias utilizadas:

- HTML para estrutura das páginas (home, desenvolvedores e jogo).
- CSS para estilização da interface, HUD e telas auxiliares.
- JavaScript para lógica do jogo, controle de colisões, movimentação, áudio e renderização no canvas.

Escopo:

- Página inicial com instruções de jogo.
- Página de desenvolvedores com informações dos integrantes.
- Tela principal do jogo com HUD, cronômetro, pontuação e ranking.
- Implementação de controles via teclado (setas e WASD).
- Sistema de colisão, aceleração, frenagem e curvas da pista.
- Ranking de melhores tempos (highscores).

2 DESENVOLVIMENTO

O projeto foi estruturado em três páginas principais:

- Página inicial (home.html): apresenta o jogo e explica os comandos.
- Página dos desenvolvedores (desenvolvedores.html): contém informações dos integrantes, incluindo nome, RA, e-mail e links para GitHub.
- Página do jogo (index.html): embarca o jogo, com canvas, HUD e botão de início.

Todas as páginas possuem menu de navegação e utilizam CSS externo para estilização.

Estrutura do código JavaScript

O arquivo script.js concentra toda a lógica do jogo, organizada em funções e classes para facilitar manutenção e clareza.

Assets e Configurações

- ASSETS.COLOR: define cores da pista (asfalto, faixas, areia).

- **ASSETS.IMAGE:** sprites do carro do jogador, carros inimigos, árvores, linha de chegada e céu.
- **ASSETS.AUDIO:** sons do jogo (motor, buzina, música tema, beep).

Funções Auxiliares

- **pad():** formata números com zeros à esquerda (ex.: tempo e pontuação).
- **clamp():** limita valores dentro de um intervalo (ex.: velocidade entre 0 e 200).
- **timestamp():** retorna o tempo atual em milissegundos.
- **accelerate():** calcula nova velocidade aplicando aceleração ou frenagem.
- **isCollide():** detecta colisões entre jogador e carros inimigos.
- **getRand():** gera números aleatórios.
- **randomProperty():** escolhe uma propriedade aleatória (ex.: faixa da pista).
- **drawQuad():** desenha quadriláteros no canvas (pista, faixas, areia).
- **sleep():** cria atrasos para contagem regressiva antes da corrida.

Classes

- **Line:** representa segmentos da pista.
 - **project():** converte coordenadas 3D em 2D para renderização.
 - **drawSprite():** desenha árvores, carros inimigos ou linha de chegada.
- **Car:** representa carros inimigos, com posição e faixa.
- **Audio:** gerencia sons.
 - **play():** reproduz sons ajustando pitch conforme velocidade.
 - **load():** carrega arquivos de áudio.

Variáveis Globais

- **highscores:** armazena melhores tempos.
- **lines:** segmentos da pista.
- **cars:** carros inimigos.
- **speed, playerX, pos, scoreVal, countDown:** estado do jogador.

Mapa da pista

- **genMap():** gera seções da pista com curvas e elevações aleatórias.
- Adiciona a linha de chegada como seção especial no final.

Controles

- W → acelerar
- S → frear
- A → virar à esquerda
- D → virar à direita

- Enter → iniciar jogo
- Escape → resetar para tela inicial
- M → ligar/desligar som

Loop Principal (update)

- Atualiza posição, velocidade e pontuação.
- Aplica aceleração, frenagem e limites de pista.
- Detecta colisões com carros inimigos.
- Renderiza pista, faixas, árvores, carros e linha de chegada.
- Atualiza HUD (tempo, pontuação, velocidade, volta).
- Controla fim de jogo (tempo esgotado ou chegada).

Inicialização

- reset(): retorna jogo ao estado inicial.
- updateHighscore(): atualiza ranking de melhores tempos.
- loadImages(): carrega sprites e chama init().
- init():
 - Configura dimensões do jogo.
 - Inicializa áudio e carros inimigos.
 - Cria segmentos da pista.
 - Configura HUD e ranking.
 - Inicia o game loop com requestAnimationFrame.

Observação importante: Durante o desenvolvimento, utilizamos ChatGPT como apoio para corrigir erros de lógica no JavaScript, especialmente em funções de colisão, renderização da pista e ajustes no loop principal. O uso foi complementar, sem substituir a compreensão do grupo sobre o código.

3 RESULTADOS

O jogo implementa de forma funcional:

- **Interface visual:** HUD com tempo, pontuação, velocidade e ranking.
- **Controles:** movimentação lateral (A/D ou setas), aceleração (W/seta para cima), frenagem (S/seta para baixo).
- **Mecânica de corrida:** pista com curvas e elevações, carros inimigos, linha de chegada.
- **Feedback sonoro:** sons de motor, buzina e contagem regressiva.
- **Ranking:** registro dos melhores tempos após cada corrida.

Testes realizados:

- Início do jogo via tecla Enter com contagem regressiva.
- Movimentação do carro e resposta aos controles.
- Colisão com carros inimigos reduzindo velocidade.
- Cronômetro funcionando corretamente.
- Registro de tempos no ranking.

Limitações observadas:

- O jogo roda apenas em navegador desktop (não otimizado para mobile).
- Ranking não é persistente (se perde ao recarregar a página).
- Colisões simplificadas (baseadas em largura aproximada).

4 CONCLUSÕES

O projeto de corrida em HTML, CSS e JavaScript atingiu os objetivos propostos, demonstrando a integração entre estrutura, estilo e lógica de programação para criar uma aplicação interativa.

Aprendizados principais:

- Uso do canvas para renderização gráfica.
- Implementação de física básica (aceleração, frenagem, curvas).
- Controle de eventos via teclado.
- Manipulação de áudio e sprites.

Possíveis melhorias futuras:

- Persistência de ranking em banco de dados ou localStorage.
- Adaptação para dispositivos móveis.
- Inclusão de níveis de dificuldade e diferentes pistas.
- Melhorias na detecção de colisão para maior realismo.

5 REFERÊNCIAS

- Documentação oficial MDN Web Docs sobre HTML, CSS e JavaScript.
- Exemplos de jogos em canvas disponíveis em CodePen e GitHub.
- Tutoriais de desenvolvimento de jogos 2D em JavaScript.
- Recursos de áudio e sprites utilizados no projeto.