Politécnico do Porto

Escola Superior de Media Artes e Design

Bruno Miguel Almeida da Justa – 9180155

Nuno Francisco Azevedo Gomes – 9180580

TETRIS

Licenciatura em Tecnologias de Sistemas de Informação para a Web

Animação Gráfica

Orientação: Professora Teresa Cristina de Sousa Azevedo Terroso

Vila do Conde, janeiro de 2020

ÍNDICE

INTRODUÇÃO

Este trabalho surgiu no âmbito da unidade curricular de Animação Gráfica, tendo como objetivo o desenvolvimento de um Tetris com gráficos 2d utilizando conceitos consolidados durante as aulas, tecnologias de animação do *HTML5*, *JavaScript*, *Canvas*, *SVG* e *CSS*.

Tetris é um dos jogos eletrónicos mais populares a nível mundial, desenvolvido por Alexey Pajitnov, Dmitry Pavlovsky e Vadim Gerasimov, e lançado em junho de 1984, foi um dos primeiros itens de exportação de sucesso da União Soviética e dos primeiros videojogos a ser visto como um vício.

Nesta entrega final, pretende-se mostrar a aplicação finalizada, as suas funcionalidades e o processo que levou à realização das animações.

APLICAÇÃO

1. Funcionalidades

A nossa aplicação tem uma página inicial (figura 1) onde são apresentados dois modos de jogo para o utilizador escolher, o modo *single player* e o modo *multiplayer*.

Se o Utilizador escolher o modo *single player* é redirecionado para a página de jogo *single player* (figura 2), onde lhe é pedido um nome de jogador. Nesta página pode-se jogar o Tetris utilizando as setas do teclado para esse mesmo efeito. A seta para cima roda a peça, a seta para baixo aumenta a velocidade de queda e as setas esquerda e direita movimentam a peça para a esquerda e direita respetivamente.

Aqui os utilizadores podem também voltar ao menu inicial, assim como ver o seu nome de jogador, pontuação e linhas que já foram feitas. Assim que o jogo termina é apresentada a página de *game over* (figura 3) onde o utilizador tem a oportunidade de jogar outra vez, dando assim *reset* ao jogo.

Por outro lado, se o utilizador escolher a opção de *multiplayer* é redirecionado para a página *multiplayer* (figura 4) onde neste caso é pedido o nome de ambos os jogadores. O jogador 1 joga com as teclas W que roda a peça, S que aumenta a velocidade de queda e A e D que movimentam a peça para a esquerda e direita respetivamente. O jogador 2 joga com os mesmos controlos que no modo *single player*.

Em comparação com o modo *single player* também é possível regressar ao menu inicial, ver os nomes dos jogadores e respetivas pontuações e linhas. Assim que um jogador perder a página de game over (figura 5) aparece mostrando o jogador vencedor e possibilitando aos jogadores jogar outra vez.

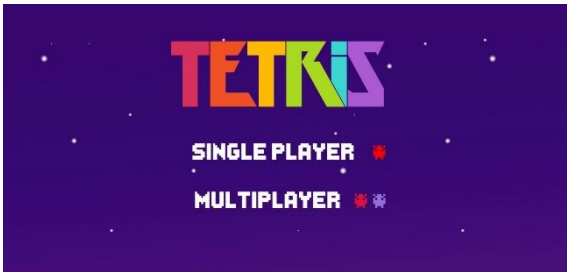


Figura 1 - Página Inicial

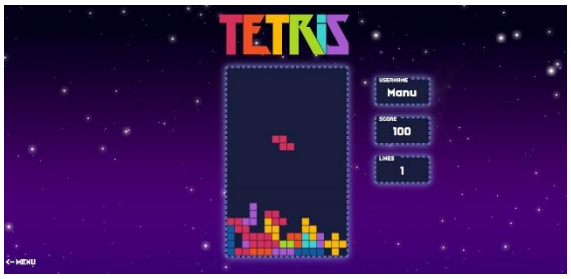


Figura 2 - Single Player

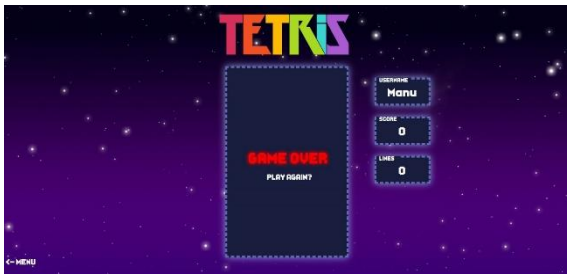


Figura 3 - Single Player Game Over

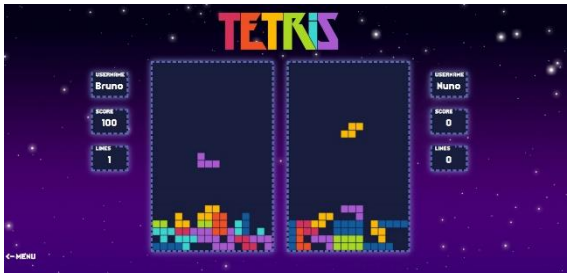


Figura 4 - Multiplayer

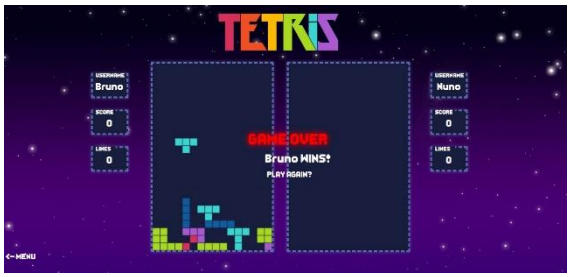


Figura 5 - Multiplayer Game Over

2. Animações

2.1. Animação do Fundo *"smoothTransition"*

Para esta animação foi atribuído um gradiente linear ao fundo do SVG com tamanho superior ao do tamanho da tela. Este gradiente vai alterando a sua posição ao longo de 12 segundos movendo-se de 50% em 50% da sua posição inicial. Para um efeito constante nesta animação atribuímos-lhe a propriedade *infinite*.

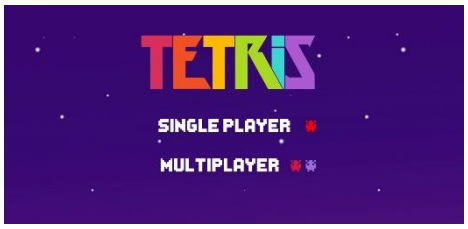


Figura 6 - Gradiente a 0%

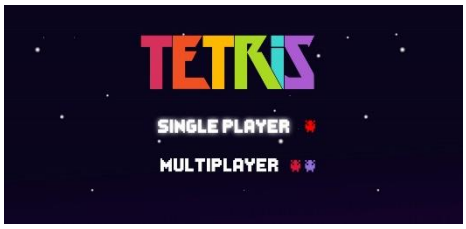


Figura 7 - Gradiente a 100%

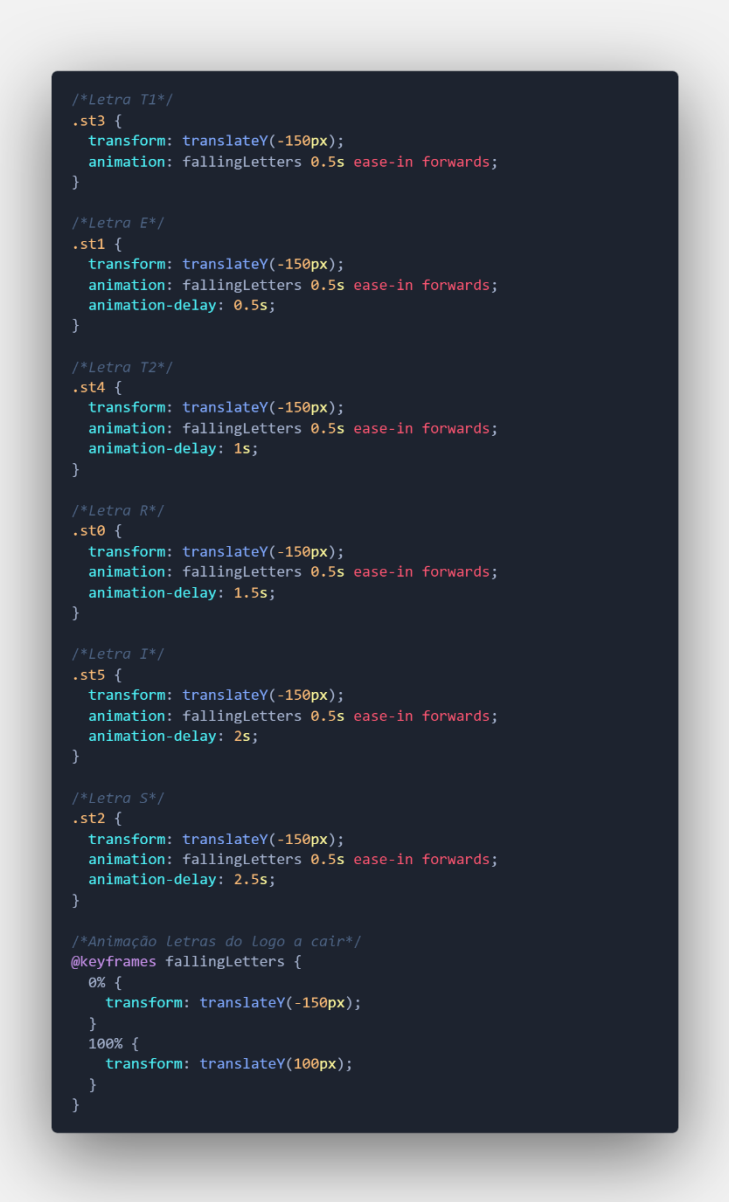
```css
/* Svg Background */
#backgr {
  animation: smoothTransition 12s ease infinite;
  background: linear-gradient(356deg, rgb(136, 41, 145),#360a73, #3f0964,#0c061e, #0c061e);
  background-size: 600% 600%;
  margin-top: -624px;
  position: absolute;
  z-index: -3;
}

/* Animaçao do Svg Background */
@keyframes smoothTransition {
  0% {
    background-position: 51% 0%
  }
  50% {
    background-position: 50% 100%
  }
  100% {
    background-position: 51% 0%
  }
}
```

Figura 8 - Código CSS da Animação do Background

2.2. Animação do Título *"fallingLetters"*

Para esta animação definimos a posição inicial de cada letra como *-150px* de modo a que não apareça na tela e alteramos essa posição para *100px* ao longo de 0.5 segundos de modo a que parece que está a cair. Definimos um delay que vai sempre aumentando 0.5 segundos de letra para letra de modo a que caia uma de cada vez. Por último definimos a prioridade *forwards* para que no fim da animação as letras se mantenham na posição final.



```css
/*Letra T1*/
.st3 {
  transform: translateY(-150px);
  animation: fallingLetters 0.5s ease-in forwards;
}

/*Letra E*/
.st1 {
  transform: translateY(-150px);
  animation: fallingLetters 0.5s ease-in forwards;
  animation-delay: 0.5s;
}

/*Letra T2*/
.st4 {
  transform: translateY(-150px);
  animation: fallingLetters 0.5s ease-in forwards;
  animation-delay: 1s;
}

/*Letra R*/
.st0 {
  transform: translateY(-150px);
  animation: fallingLetters 0.5s ease-in forwards;
  animation-delay: 1.5s;
}

/*Letra I*/
.st5 {
  transform: translateY(-150px);
  animation: fallingLetters 0.5s ease-in forwards;
  animation-delay: 2s;
}

/*Letra S*/
.st2 {
  transform: translateY(-150px);
  animation: fallingLetters 0.5s ease-in forwards;
  animation-delay: 2.5s;
}

/*Animação Letras do logo a cair*/
@keyframes fallingLetters {
  0% {
    transform: translateY(-150px);
  }
  100% {
    transform: translateY(100px);
  }
}
```

Figura 9 - Animação fallingLetters

2.3. Animação Aparecimento das Opções *"fadeIn"*

Para a animação de aparecimento das opções apenas definimos uma opacidade inicial de 0 que ao longo de 1.2 segundos vai mudando a opacidade para 1. Demos um delay de 2.8 segundos para que a animação comece quando a animação *fallingLetters* esteja a terminar. Atribuímos também a propriedade *both* para que no início da animação a opacidade das opções seja 0 e no fim seja 1.

```css
/*Container com as Opções*/
#buttons {
  animation: fadeIn 1.2s both;
  animation-delay: 2.8s;
}

/*Animação das opções a aparecer*/
@keyframes fadeIn {
  0% {
    opacity: 0;
  }
  100% {
    opacity: 1;
  }
}
```

Figura 10 – Animação fadeIn

2.4. Animação *Hover* das Opções *"vibration"*

Para o *hover* das opções decidimos atribuir uma sombra com cor branca de modo a que pareça um brilho idêntico ao das estrelas do fundo. Criamos também a animação *vibration* para que a opção selecionada se mova de modo a dizer ao utilizador que é a opção selecionada. Nesta animação vamos alterando as posições de x e y da opção selecionada que vão alterando de 30% em 30% de um modo constante.

```css
/*Opções de Single Player e Multiplayer*/
#playTxt:hover {
  text-shadow: 0 0 18px white;
  animation: vibration 0.3s linear infinite both;
}

/*Animação das opções a vibrar*/
@keyframes vibration {
  0% {
    transform: translate(0);
  }
  30% {
    transform: translate(-2px, 2px);
  }
  60% {
    transform: translate(2px, 2px);
  }
  90% {
    transform: translate(2px, -2px);
  }
  100% {
    transform: translate(0);
  }
}
```

Figura 11 – Animação Vibration

2.5. Animação dos Bonequinhos *"jello"*

Nesta animação usamos o scale3d de modo a que pareça que os bonequinhos se estão a mexer tipo gelatina, esta animação acontece ao longo de 2 segundos de uma forma repetitiva.

```css
/*Bonequinhos*/
#bitGuy {
  animation: jello 2s ease-in-out infinite both;
}

/*Animação dos bonequinhos a mexer*/
@keyframes jello {
  0% {
    transform: scale3d(1, 1, 1);
  }
  30% {
    transform: scale3d(1.25, 0.75, 1);
  }
  40% {
    transform: scale3d(0.75, 1.25, 1);
  }
  50% {
    transform: scale3d(1.15, 0.85, 1);
  }
  65% {
    transform: scale3d(0.95, 1.05, 1);
  }
  75% {
    transform: scale3d(1.05, 0.95, 1);
  }
  100% {
    transform: scale3d(1, 1, 1);
  }
}
```

Figura 12 – Animação jello

RECURSOS

Para uma boa elaboração deste projeto o grupo considerou essencial a utilização do seguinte software:

- Adobe Photoshop;
- Adobe Illustrator;
- Visual Studio Code;
- Imagem de Fundo: funlava.com/wp-content/uploads/2013/09/Purple_Space_Background_by_YuniNaoki.jpg
- Logo: deviantart.com/jmk-prime/art/Tetris-Logo-512366364~
- Animador de Gradiente: gradient-animator.com/

CONCLUSÃO

Durante o desenvolvimento da aplicação, foram surgindo problemas e desafios, para os quais nem sempre arranjamos solução. No entanto maior parte deles acabaram por ser superados com algumas dificuldades.

O trabalho desenvolvido serviu como motor de melhoramento de conhecimentos de *Canvas*, *SVG* e *CSS* assim como da plataforma Visual Studio Code.

Desta forma, este projeto foi um desafio cativante, pois conseguimos aplicar e adquirir novos conhecimentos essenciais para o nosso futuro enquanto profissionais da área.