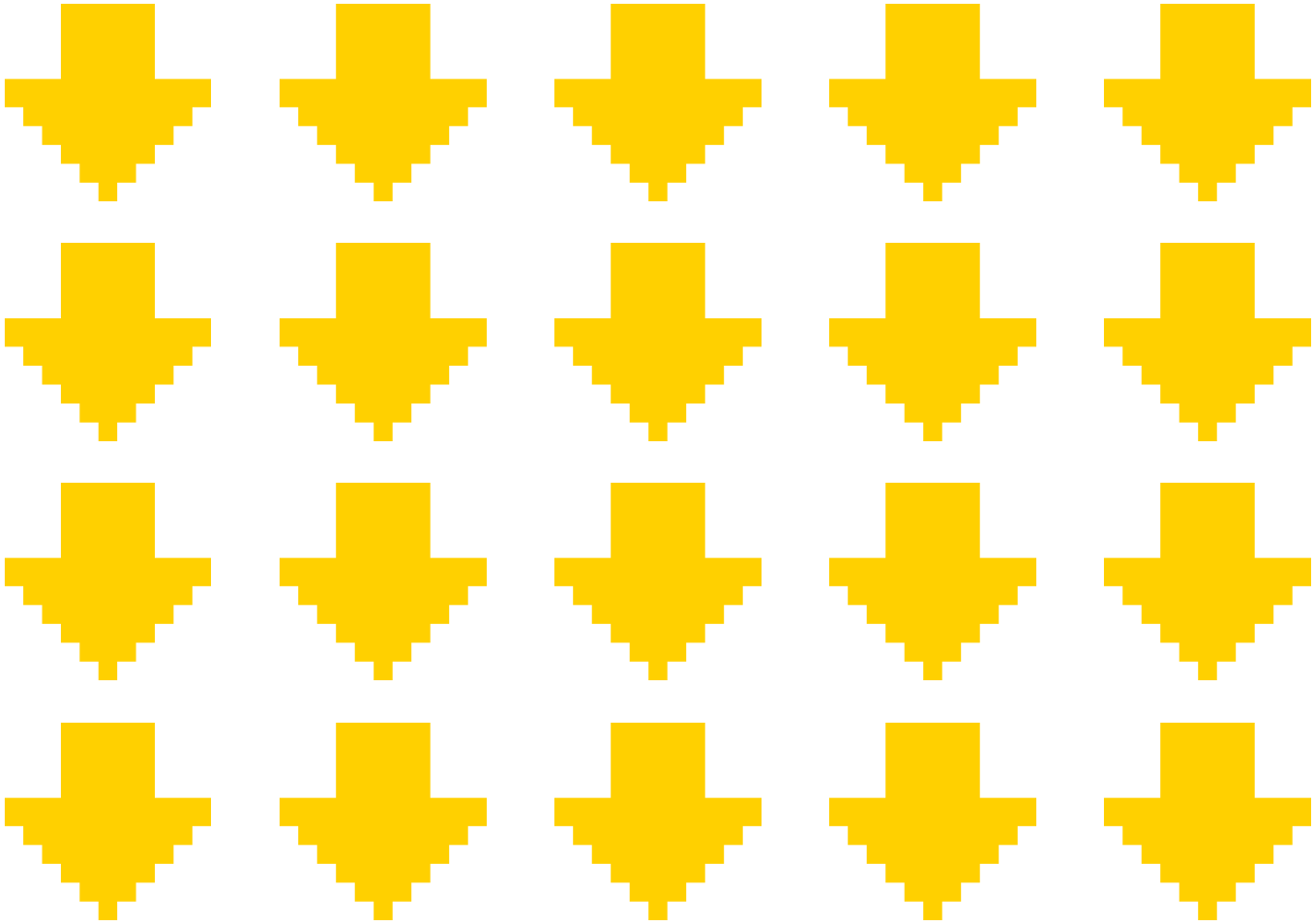
FACULDADE DE BELAS-ARTES DA UNIVERSIDADE DO PORTO

P2 SEQUENCIADOR AUDIOVISUAL

STAR WARS
PAMADO ADVENTURES



Arthur Gomes | up201902247
Cristiana Braz | up201906259
Erika de Ornelas | up201904934
Yéssica Josie | up201904956

17/06/2021 Porto, Portugal
Laboratório de Som e Imagem (LSI202)

ÍNDICE

SINOPSE

Este trabalho trata-se da criação de um sequenciador audiovisual através da aplicação dos conceitos de programação apreendidos durante as aulas de LSI (Laboratório de Som e Imagem), integrando os vários meios digitais estáticos, como imagens, texto e gráficos, e os meios dinâmicos, tais como vídeos, sons e animações, de forma a criar um projeto interativo a partir do uso do teclado e do rato do computador.

Star Wars: Pamado Adventures é um jogo do estilo *infinity run* protagonizado pelo professor Pedro Amado, inserido no universo dos clássicos filmes do Star Wars. O personagem, equipado com um jetpack, percorre um trajeto com obstáculos que reagem à música de fundo, enquanto que diferentes objetos e personagens bónus passam pelo ecrã, em variados intervalos. Para além das músicas de fundo, o jogo contém também diversas linhas de áudio gravadas durante as aulas do semestre que foram selecionadas e implementadas de modo a responderem às várias possíveis situações dentro do jogo. O objetivo é obter o maior score possível, sendo que quanto mais tempo o jogador se conseguir esquivar dos obstáculos e não perder as suas vidas, maior será a sua pontuação final.

PALAVRAS-CHAVE:
infinity run; side-scroller; star wars; pixel art; audio lines

INTRODUÇÃO

TEMA/CONCEITO/OBJETIVOS

Utilizando o tema do universo do Star Wars, o projeto desenvolvido é um jogo infinito do tipo *side-scroller*. Tínhamos como objetivo aplicar todos os conhecimentos obtidos durante as aulas deste semestre, cumprindo o enunciado do trabalho e todos os parâmetros nele especificados. Assim, decidimos criar um jogo com elementos específicos em homenagem a tudo o que experienciamos este ano nas aulas de LSI. Desta forma, escolhemos o tema do Star Wars, adaptando os nossos personagens, obstáculos e objetos a este universo, tanto a nível gráfico como a nível conceptual. Quanto ao conceito do projeto, decidimos elaborar um jogo inspirado nas arcades retro, optando por fazer algo mais apoiado na jogabilidade e nos gráficos antigos, nomeadamente na *pixel art*.

Para a conceção do jogo em si, definimos os nossos objetivos de uma forma mais direcionada e objetiva, compilando uma lista com todas as coisas que queríamos implementar. Tínhamos como finalidade assim, a criação de um jogo infinito orientado na horizontal, a elaboração de várias telas dentro do jogo (criando um menu principal, uma tela com as instruções, com os créditos, e outra com a história do jogo), a implementação de sons e falas retiradas das aulas gravadas durante o semestre como forma de interação entre o personagem principal e os restantes objetos ao seu redor, o desenvolvimento de gráficos originais que se mantivessem coesos e em concordância no projeto inteiro, e no geral a elaboração de uma aplicação interativa que desse liberdade ao jogador. Outro dos objetivos principais era incluir, de um modo bem-humorado, o professor Pedro Amado no universo de Star Wars, bem como homenageá-lo pelo seu modo de ensino, sempre de forma dinâmica, divertida e enérgica. É um exercício metalinguístico, no qual usamos o próprio motivo da aula como execução do trabalho final da disciplina.

IMPORTÂNCIA DO CONTEXTO

Para além dos objetivos anteriormente referidos, decidimos desenvolver este jogo com o propósito de aliviar e retirar um bocado da seriedade nos projetos efetuados dentro do contexto académico. Com muitos dos trabalhos realizados nas diversas disciplinas do semestre, notamos a crescente austeridade que lhes era atribuída, no sentido de que tudo tinha um significado e propósito, tudo aparentava ter de ser justificado com algum tipo de explicação sentimental, filosófica ou conceitual, não deixando espaço para a exploração de ideias que os alunos gostariam de perseguir simplesmente por ser algo que lhes interessava. Com a observação da necessidade de fundamentar todos os aspetos de um trabalho, decidimos contrariar esta noção criando o nosso jogo com o intuito de produzir algo divertido motivado pelos nossos interesses pessoais (mas que cumprisse o enunciado da proposta ao mesmo tempo). Embora percebamos que o Design vive da comunicação e da transmissão de ideias, foi do nosso interesse contornar um bocado este ideal e esta “regra” que tem vindo a ser estabelecida, dando vida a um projeto que nasceu da vontade de 4 alunos, e não propriamente de um conceito fundamentado por uma explicação complexa.

DANIEL SHIFFMAN

Daniel Shiffman trabalha como Professor de Artes no Programa de Telecomunicações Interativas da *Tisch School of the Arts* da *NYU*. Originalmente de Baltimore, Daniel recebeu um *BA*, *Bachelor of Arts*, em Matemática e Filosofia da Universidade de Yale e um mestrado do *ITP*. É diretor da *The Processing Foundation* e desenvolve tutoriais, exemplos e bibliotecas para *Processing* e *p5.js*.É o autor de *Learning Processing: A Beginner's Guide to Programming Images, Animation, and Interaction* e *The Nature of Code*, um livro de código aberto sobre a simulação de fenómenos naturais em *Processing*. Shiffman tem a missão de compartilhar a programação com o mundo (de uma forma divertida e acessível). Pode ser encontrado a falar incessantemente sobre programação no seu canal do YouTube, *The Coding Train*, onde publica tutoriais de "codificação criativa" com assuntos que vão desde o básico de linguagens de programação como *JavaScript* (com *p5.js*) e *Java* (com *Processing*) até algoritmos generativos como simulações de física, visão computacional e visualização de dados.

Para o nosso projeto, os vídeos no seu canal de youtube foram de grande ajuda, sendo que várias partes do nosso código foram desenvolvidas a partir das explicações e dos exemplos do Shiffman.

Img1. Daniel Shiffman, The Coding Train, 2017.

MARIO GAME DE MIKE "POMAX" KAMERMANS

Durante o processo de recolha de referências deparamos-nos com um link que nos direcionou para uma página com a explicação de como fazer um jogo do Super Mario passo a passo. Após uma maior inspeção, descobrimos o seu criador, Mike Kamermans. Sendo que o jogo do Mario descrito e explicado neste site partilhava de várias ações e conceitos que queríamos explorar e implementar no nosso próprio jogo, consideramos este projeto como uma das referências que mantivemos em mente, consultando o código nele desenvolvido e comparando-o com o nosso trabalho, adaptando ideias e linhas de código às nossas necessidades.

Disponível em: http://processingjs.nihongoresources.com/test/PjsGameEngine/docs/tutorial/mario.html

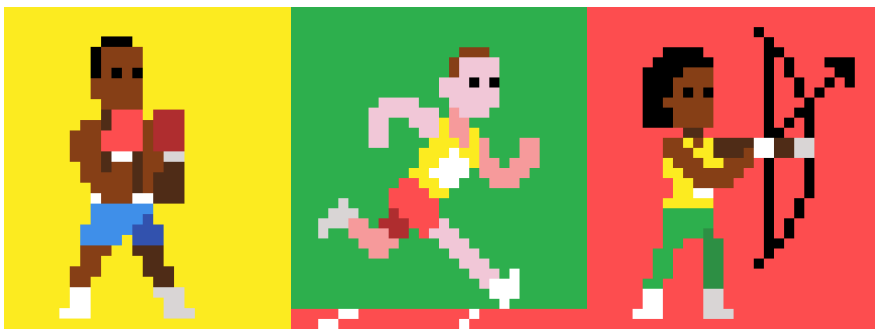
STAR WARS: TINY DEATH STAR

Star Wars: Tiny Death Star é um videojogo de simulação de negócios desenvolvido pela *Disney Mobile* e *NimbleBit*. Foi baseado no jogo anterior da *NimbleBit, Tiny Tower*, e foi ambientado no universo de Star Wars. O objetivo é construir e expandir uma Estrela da Morte, atraindo pessoas virtuais conhecidas como Bitizens galácticos para construir e administrar negócios virtuais dentro da Estrela. Para o nosso projeto conduzimos uma pesquisa sobre jogos e ilustrações baseadas na *pixel art* que se incluíssem no mundo e no tema do Star Wars de forma a termos uma ideia do que já existia. Assim encontramos o jogo *Star Wars: Tiny Death Star* que utilizamos como referência para as nossas ilustrações, e para percebermos como personagens e objetos dos filmes originais poderiam ser adaptados a este tipo de desenho.

Img2. Star Wars: Tiny Death Star, 2013.

IVAN DIXON

Outra referência a nível gráfico foi Ivan Dixon, um dos *pixel artists* mais bem conhecidos atualmente. Dixon é diretor de animação e do estúdio *Showoff*, ilustrador e designer conhecido pelo seu estilo de arte pixel e tradicional. Produzindo conteúdo para uma variedade de clientes, incluindo videoclipes para Childish Gambino, Surfaces, Elton John e Gotye, entre outros, já fez também sequências de títulos em pixels para *The Simpsons*, *Rick & Morty* e *Adventure Time*. Escolhemos referenciar as ilustrações mais simples do artista, tirando inspiração das paletes de cores vibrantes e do nível de detalhe, que apesar de não ser exagerado, é o suficiente para retratar de forma fiel as suas personagens.



Img3. RETROLYMPIAD, Ivan Dixon, 2016.

PRODUÇÃO

Relativamente ao código, começamos por utilizar como base o tutorial do *Chrome Dinosaur game*, do Daniel Shiffman, devido ao facto de que a sua jogabilidade se enquadra bastante bem à parte inicial do nosso conceito, uma personagem principal controlada no eixo vertical, e diversos objetos que entram e saem da tela de jogo. O primeiro passo foi a criação destes elementos em duas classes diferentes. Ao longo de todo o código optamos por trabalhar através de **ficheiros.js** separados, nomeadamente, classes e funções que assumem o papel de telas de jogo, ao invés de os definir no **sketch**, que ficou reservado apenas para a inicialização e definição de variáveis. Começamos por definir as suas propriedades e coordenadas, assim como variáveis específicas para cada um, no caso do Pamado, o protagonista, uma força gravitacional que o puxe ao chão, e no caso dos obstáculos, a velocidade em que se mexem (no eixo horizontal), e as respetivas funções para a sua execução. A variável **this.altura** foi importante, não só para definir a área de movimento, como a área de saída de todos os elementos utilizando o **constrain**, que limita um valor.

Para pô-los em ação foi necessário o uso de diversas condições, como **if (keyIsDown(____)){**, para mudar as coordenadas do Pamado. Por outro lado, a condição para a criação de obstáculos foi diferente. De modo a que o nosso projeto se enquadrasse melhor à proposta, bem como para torná-lo mais dinâmico e conferir-lhe maior ligação à música, fizemos com que estes objetos (os asteroides) fossem desenhados de acordo com a amplitude do som, em determinados momentos da música.

Para acrescentar valor gráfico ao jogo, desenvolvemos vários GIFs. Os quais carregamos no código a partir do método **createImg**; utilizando as funções **.hide( )** e o **show( )** para ter um maior controlo de onde aparecem os GIFs e evitar a sua visualização durante o "Loading…". Para conseguir simular o avanço da nossa personagem no espaço, foi necessária a criação de uma imagem de fundo em movimento em loop contínuo. Conseguimos obter este efeito desenhando duas imagens, uma a ocupar a totalidade do ecrã, e outra a começar na largura final do ecrã, ou seja, "fora" do espaço de visualização. Com isto, ao criar o movimento das imagens, sempre que uma começava a "sair" do ecrã, a que estava do lado de "fora" entrava no espaço de visualização, isto tudo em loop contínuo, criando assim a ilusão de uma imagem de fundo infinita.

Posteriormente adicionamos duas classes novas, **Nave**, outro objeto que o Pamado deve evitar, e **Bónus**, constituída por 3 objetos, cada um com efeitos diferentes.
Uma vez que a coordenada **x** de todos os objetos que se movimentam no eixo horizontal seja menor do que 0 menos a sua largura, ou seja, quando sai do ecrã, executa **splice** a partir de um ciclo definido para cada classe.

```
375   //Bónus (Miffy, Baby Yoda)
376   for (let i = 0; i < bonus.length; i++) {
377     if (bonus[i].x < 0 - bonus[i].w) {
378       bonus.splice(i, 1);
379     } else if (screen == 6) {
380       // quando for a tela de gameover, "limpa" os bónus
381       bonus.splice(i, 5);
```

Img4. Printscreen do código, splice, 2021.

Outra das questões principais a desenvolver para cumprir o conceito geral do jogo foram as colisões. Visto que a nossa personagem teve de ser capaz de perder vidas, assim como ter uma motivação para as arriscar, decidimos incluir elementos bónus. Por influência do Shiffman, decidimos dar uso da biblioteca **p5.collide2D**. Basicamente funciona como uma ferramenta que calcula a deteção de colisões entre figuras geométricas, através dos parâmetros inseridos; devolve (returns) o resultado verdadeiro ou falso desse cálculo. No nosso caso utilizamos especificamente a **collideCircleCircle**, de forma a conseguir inserir melhor as imagens dentro das elipses, e dar compensações, de forma a não fazer **hits** injustos e tornar chata a experiência de jogo. É importante destacar a criação da função **hits( )**, dentro da classe Pamado. Se o resultado do **collideCircleCircle** é verdadeiro, **hit** (variável definida como "false" em todas as classes que podem colidir com Pamado) passa a ser igual a "true".

```
12    // DEFINIR HIT
13    hits(obs) {
14      let x1 = this.x + this.w * 0.5;
15      let y1 = this.y + this.h * 0.5;
16      let x2 = obs.x + obs.w * 0.5;
17      let y2 = obs.y + obs.h * 0.5;
18      //valores multiplicados por 0.5 devido a que a função
    da biblioteca (collideCircleCircle) pede o valor do
    diâmetro, mas no nosso caso temos dois valores diferentes
    (w,h)
19
20      ellipseMode(CENTER);
21      return collideCircleCircle(x1, y1, ((this.w || (this.h
    - 10))), x2, y2, ((obs.w || obs.h) - 30)) ; // ----
```

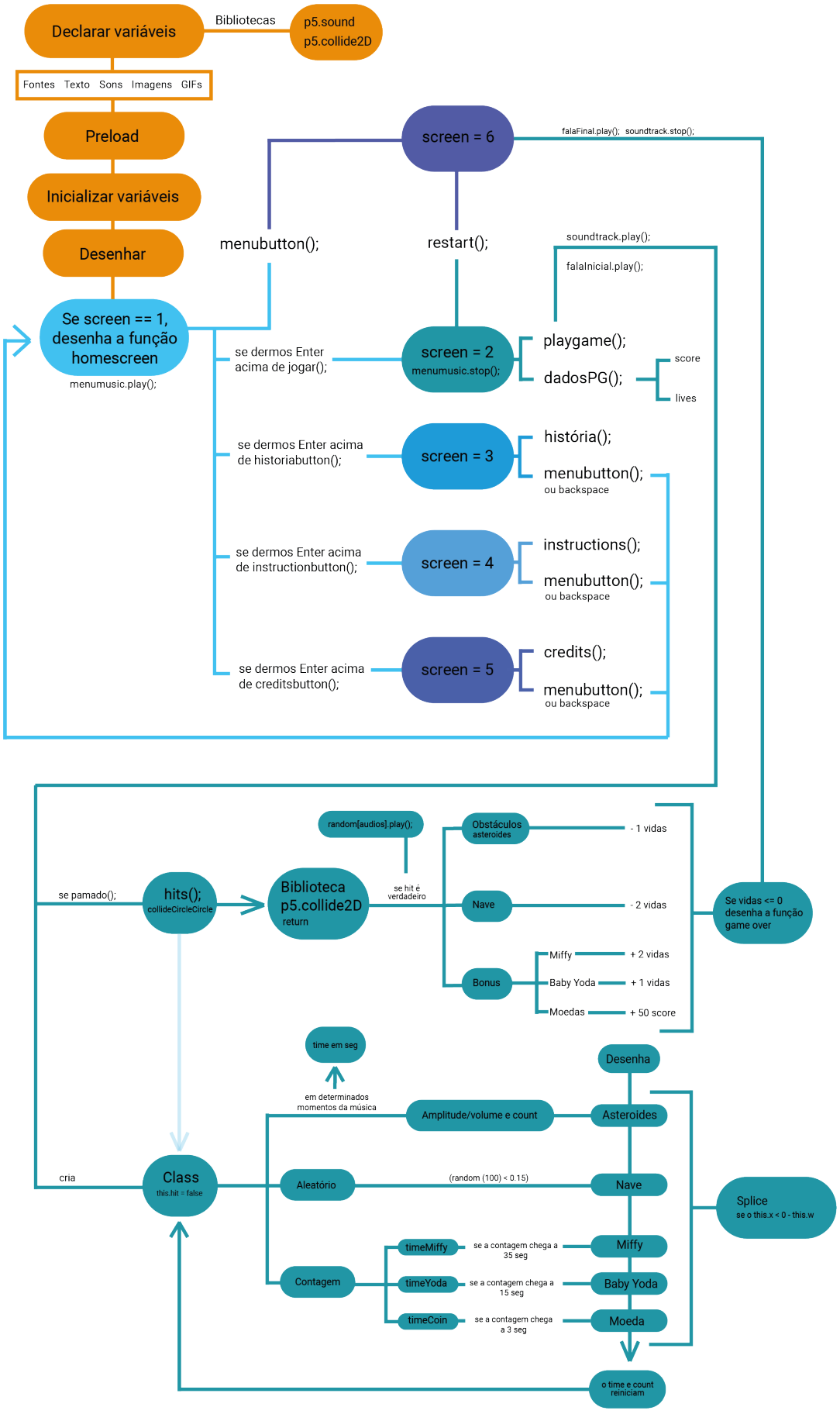
Img5. Printscreen do código, hits, 2021.

Foi necessária a criação de duas variáveis que contabilizem as vidas e o score, pois a colisão com cada um dos objetos tem um efeito diferente sobre elas. Se não for possível evitar os obstáculos, asteroides e naves, perde-se uma ou duas vidas respetivamente. Em contrapartida, apanhando um bónus ganha-se vidas de volta, uma com o Baby Yoda e duas com a Miffy, assim como no caso de apanhar uma moeda, recebe-se 50 pontos para o score. Desta forma, utilizamos o **pamado.hits( )** dentro de diferentes **ifs** para ativar estes acontecimentos, incluindo também a reprodução de áudios específicos, como as falas do professor e efeitos sonoros. As linhas de fala foram agrupadas em **arrays** segundo as situações em que são reproduzidas, sendo que assim podem ser escolhidas de forma aleatória pelo uso do random. No que diz respeito à interface do jogo, precisamos de criar uma variável, **screen**, que identifica as nossas 6 telas, assim como uma função específica para cada uma delas, as quais são chamadas no **draw**, se a condição referente a elas é cumprida. Dentro das funções, **homescreen( )**; e **playgame( )** são reproduzidas as suas respetivas músicas de fundo. Por último, adicionamos ainda uma função **restart**, que devolve as variáveis inicializadas dentro dela ao seu valor inicial.

```
423 //RESTART.....................................
424 function restart() {//<--- quando é chamada, esta função
    devolve estas variáveis ao seu valor inicial, simulando um
    restart do jogo
425
426
427   lives = 3;
428   score = 0;
429   count = 10;
430   time = 0;
431   timeFalaOver = 0;
432   timeFala = 0;
433   speed = 5; //mudar a velocidade aqui
434   amp = new p5.Amplitude();
435   cantina.jump(0);
436   screen = 2;
437   forcespirit.hide();
```

Img6. Printscreen do código, restart, 2021.

DIAGRAMA

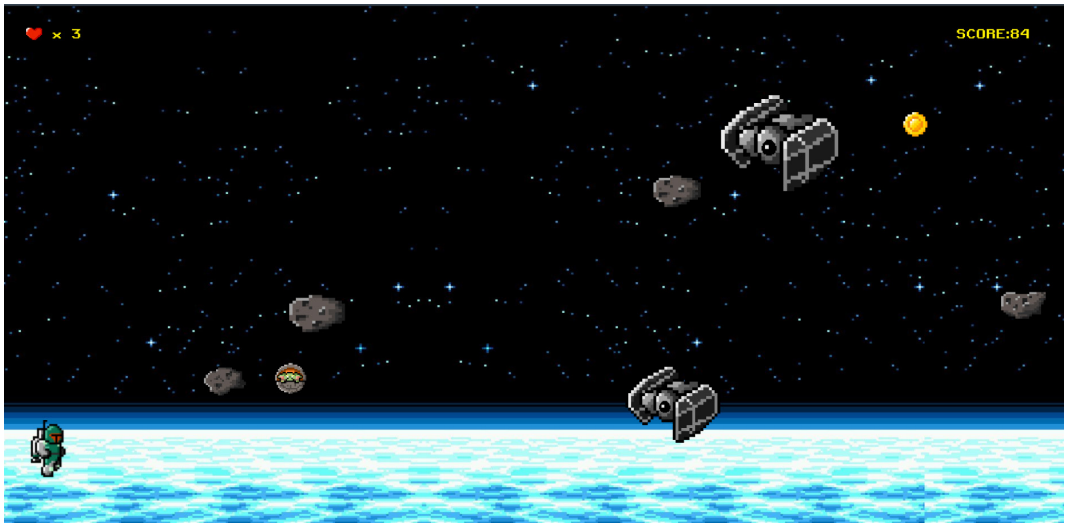


Img7. Diagrama explicativo do código, 2021.

JOGABILIDADE

Após a explicação de como o código foi desenvolvido, é importante explicar como o jogo funciona ao pôr o código em prática. O jogo usa um sistema simples com as setas (para cima e baixo) para controlar o jetpack. Quando o jogador não pressiona nenhuma tecla o personagem cai lentamente, sob o efeito leve da gravidade. Por estar continuamente em movimento, o jogador não controla a sua velocidade, apenas o seu movimento ao longo do eixo vertical.

O objetivo do jogo é chegar o mais longe possível, recolher os bónus para aumentar as vidas e evitar perigos como os asteróides e naves espaciais. O contacto com qualquer um desses obstáculos resulta na perda de uma vida. O jogador inicia o jogo com três vidas, mas consegue acumular mais caso consiga apanhar uma Miffy ou um Baby Yoda, sendo que consegue apanhar também moedas douradas para aumentar o score total. Ao morrer, o jogador pode verificar o seu score e reiniciar o jogo. Conta também com uma tela de instruções para verificar os controles e conhecer os obstáculos e bónus. Além disso, o menu contém uma tela de história, que apresenta o personagem e explica a sinopse do jogo, no estilo de opening text do filme Star Wars.



Img8. Printscreen de gameplay de Star Wars Pamado Adventures, 2021.

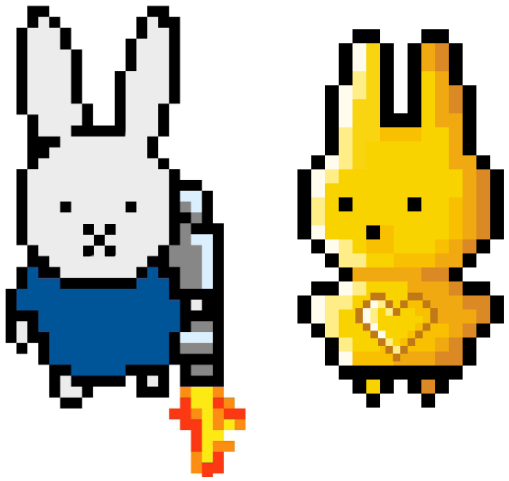
ÁUDIO

Ao todo, 60 linhas de áudio estão incluídas no jogo, sendo que foram compiladas das aulas que foram ensinadas ao longo do semestre. Estes áudios estão separados em diversas categorias, que reagem aos eventos que ocorrem no jogo como, por exemplo, na colisão com obstáculos e na recolha de itens especiais, ou seja, nas decisões que o jogador toma. Cada categoria contém várias opções, para que os áudios não se tornem repetitivos. Desta forma, sempre que acontece uma colisão com um obstáculo ou um elemento bónus, o personagem Pamado é capaz de ”falar”, reproduzindo um dos áudios direcionados a esse acontecimento específico. Para além das falas da persongem e de outros efeitos sonoros (como o som do jetpack e do andar da persongem), durante o percurso do jogo são reproduzidas três músicas em loop: *Cantina Star Wars Theme*, The *Mandalorian Theme* e *Imperial March song*.

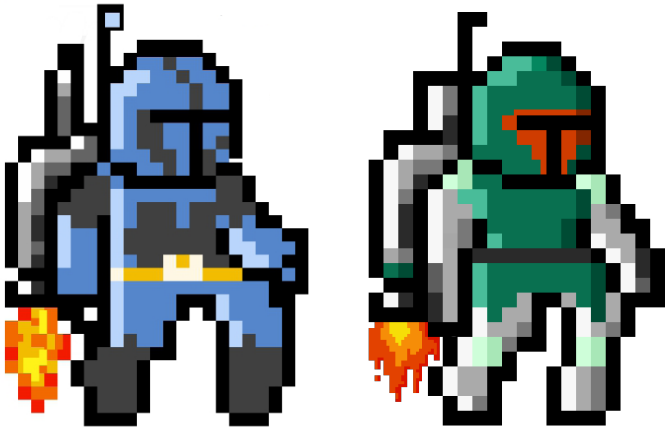
GRÁFICOS

Seguindo o conceito das arcades retrô, as ilustrações e toda a estética do trabalho foi pensada para relembrar os jogos antigos, com gráficos apoiados no estilo da pixel art e sons e músicas no estilo 8-bit. Com o tema do Star Wars, decidimos incluir a nossa personagem principal no povo dos Mandalorian, traduzindo isso para a sua ilustração. Para os obstáculos, 5 tipos diferentes de asteroides foram criados, variando maioritariamente nas suas formas e nos seus respetivos tamanhos. Para a criação de um tipo de obstáculo mais perigoso e difícil de contornar, foi desenvolvida a ilustração de uma nave (TIE fighter) muito reconhecível e muito característica dentro do universo de Star Wars. Para os elementos bónus, decidimos representar também um personagem atualmente facilmente identificável, o Baby Yoda, aceitando o desafio de o tentar retratar no estilo pixelizado sem perder muito detalhe num tamanho minimizado. Em contraste com a fama deste último elemento, e como forma de trazer mais algo das aulas de LSI para o projeto, foi adicionada uma Golden Miffy como um outro bónus que representa uma espécie de *inside joke* partilhada no decorrer do semestre, sendo ela ilustrada também em concordância com os outros elementos. De maneira a introduzir mais dinamismo e variedade no personagem do Pamado, este foi animado numa série de frames em loop contínuo, criando a animação das chamas do seu jetpack e o correr da própria personagem. Outras versões foram também ilustradas, nomeadamente a forma de *Force Ghost* que aparece quando o jogador fica sem vidas e acontece o *Game Over*. Todos os elementos restantes como a imagem de fundo e as moedas bónus receberam o mesmo tratamento visual e tipo de ilustração, sendo que no projeto todo foi utilizada uma fonte para os elementos escritos que remetesse para o conceito dos jogos antigos.

Podemos observar os alguns dos elementos desenvolvidos, em como os que sofreram maiores alterações, nomeadamente a Miffy e a personagem principal, o Pamado.



Img9. Miffy, antes (esquerda) e depois (direita), 2021.



Img10. Pamado, antes (esquerda) e depois (direita), 2021.

Img11.Obstáculos, asteroides e nave, 2021.

CONCLUSÃO

ANÁLISE DO PROCESSO E REFLEXÃO CRÍTICA

Analisando o nosso processo de trabalho, começamos logo no início por realizar uma série de reuniões entre os membros do grupo de forma a decidirmos o rumo do projeto, de maneira a garantirmos que todos se encontravam confortáveis e se mantinham informados em relação às decisões tomadas e implementadas. Estando todos em concordância e tendo definido os conceitos básicos e o tema do projeto, passamos para uma fase de recolha de referências e material útil, revendo as aulas gravadas durante o semestre, e consolidando os nossos conhecimentos sobre a matéria. Numa fase inicial optamos por utilizar a aplicação *Visual Studio Code* que nos permitiria trabalhar simultaneamente no mesmo código, possibilitando uma distribuição de tempo mais eficiente. No entanto, após algumas complicações com o funcionamento da aplicação, tivemos de abandonar a ideia e passar para uma abordagem diferente. Como tal, achamos melhor trabalhar a partir da divisão de tarefas, juntando tudo depois num código master, que foi sendo atualizado e passado entre os membros do grupo. Esta divisão de tarefas funcionou a nosso favor no sentido de que nos permitiu manter um nível de organização elevado, sendo que criamos listas e apontamentos do que era necessário fazer e do que já estava terminado. Optamos também por criar um grupo no *Discord*, o que possibilitou a nossa comunicação à distância e a fácil partilha de todo o tipo de informações.

Por outro lado, o facto de não conseguirmos trabalhar no mesmo código obrigou a existência de vários ficheiros com as partes constituintes, o que por vezes dificultou a visualização e noção do projeto como um todo. No geral conseguimos manter um ambiente de trabalho organizado e focado, sendo que no futuro gostaríamos de encontrar uma forma de tornarmos a edição do código com várias pessoas ao mesmo tempo possível.

AVALIAÇÃO DE UTILIZADORES EXTERNOS

Apresentando o nosso jogo a pessoas que não se encontravam envolvidas no desenvolvimento do projeto, recebemos várias reações parecidas e comentários minimamente variado. No geral, os utilizadores encontraram-se recetivos à ideia e ao jogo, sendo que o tema do Star Wars aparentou ser muito apreciado. A existência de um menu e várias telas a explicar as instruções e o contexto da história transmitiram aos utilizadores a sensação de um jogo bem formulado. Os gráficos em funcionamento com o estilo das músicas e dos sons comunicaram de forma eficiente a influência dos jogos antigos e de arcade. A implementação das falas do personagem em relação aos eventos que acontecem ao seu redor revelou-se como sendo um dos pontos que os jogadores mais mencionaram, considerando estes sons como um aspeto divertido e cómico. Por outro lado, outros acharam o jogo um bocado repetitivo, no sentido que os eventos que acontecem são maioritariamente sempre os mesmos. Por essa razão, comentaram também que seria interessante adicionar um aspeto de personalização ao jogo, como escolher a personagem ou haver mudanças de imagem de fundo, sendo que a existência de diferentes níveis nos quais seria possível jogar o *infinity run* foi algo também mencionado.

SIMULAÇÃO DO PROJETO

Para as aplicações, decidimos representar o nosso trabalho não só no suporte em que o jogo foi pensado inicialmente, que seria num computador portátil ou fixo, mas também numa versão adaptada a uma máquina de arcade. Adicionalmente, optamos por desenvolver um poster para propósitos promocionais.

Img12. Simulação numa máquina de arcade, 2021.

Img13. Simulação num computador portátil, 2021.

Img14. Poster promocional desenvolvido, 2021.

Img15. Simulação do poster promocional, 2021.

PROBLEMAS NO DESENVOLVIMENTO/ESTRATÉGIAS DE CORREÇÃO

Para conseguir criar um só objeto de cada vez definimos diversas variáveis para contabilizar o tempo (**count**, **timeMiffy**, **timeYoda**…) Uma vez que a condição é cumprida, os **timers** voltam a zero. Este processo é executado em forma de loop. Isto resolveu o problema da aparição de vários objetos em simultâneo, mas não sabemos se foi a forma mais eficiente ou correta de o fazer. Um dos problemas mais recorrentes ao longo de todo o processo de programação foram sem dúvidas as colisões. Pensamos que o facto de utilizar uma biblioteca facilitou algumas questões, como seria o facto de tornar a sua execução mais simples e imediata a nível de código. Porém, encontramos muitas inconveniências na colisão de objetos da mesma classe, ou seja, situações onde os **hits** com alguns dos objetos da mesma classe tinham como consequência o mesmo efeito, achamos que por alguma razão existia algum tipo de conflito. Para resolver isto, tivemos de criar duas classes diferentes de obstáculos, uma com os asteroides e outra com as naves, sendo que inicialmente eram da mesma classe. Apresentou-se a mesma situação na classe **Bónus**, por exemplo, se existissem várias moedas ao mesmo tempo no ecrã, só era detetada a última colisão. Conseguimos "resolver" esta questão provisoriamente, colocando um tipo de objeto de cada vez. Contudo, achamos que foi enriquecedor para o projeto e o nosso processo de aprendizagem utilizar as colisões como uma oportunidade para começar a explorar as bibliotecas do **p5js**.

Após a existência de alguma confusão por parte de utilizadores externos ao grupo em relação ao controlo do menu ser com o rato ou com o teclado, sendo que não é algo que esteja especificado de forma escrita, tentamos simular botões a partir de retângulos que limitassem a área de clique, para chamar as funções específicas que nos levam às diferentes telas. Contudo, isto não funcionou completamente da forma que queríamos, concentrando a área de ação do clique num espaço muito limitado. Esta é uma questão para a qual gostaríamos de encontrar uma solução no futuro. Outro problema foi o plano 3D da tela da História. Tendo de utilizar **WEBGL** para conseguir criar o espaço 3D e, consequentemente, conseguir criar a inclinação do texto, encontramos complicações a tentar conjugar o modo 3D dessa tela específica com o modo 2D das restantes telas, sendo que ultimamente tivemos de retirar o aspeto da tridimensionalidade. Gostariamos no futuro de conseguir implementar isso sem complicações e de forma funcional.

IMPLEMENTAÇÕES FUTURAS

Através das experiências obtidas no decorrer da realização deste projeto, podemos apontar a correção de certos problemas e a implementação de outros métodos de forma a melhorar a nossa dinâmica de trabalho no futuro. De modo a superar um dos maiores obstáculos com que nos deparamos, gostaríamos de arranjar maneira de conseguirmos trabalhar simultaneamente no mesmo código, com o objetivo de poupar tempo e evitar confusões entre ficheiros. Com isto, seria do nosso interesse conseguirmos desenvolver o projeto ainda mais, adicionando maior complexidade na jogabilidade, introduzindo mais ações (como derrotar inimigos, apanhar armas e veículos, por exemplo), aumentar a variedade de músicas de fundo e efeitos especiais sonoros, bem como as falas do personagem e, no geral, conseguir disponibilizar mais tempo para tornar as nossas ambições em realidade. Melhorar a gestão de tempo através da implementação de horários de trabalho semanais e reuniões entre o grupo mais frequentes será um dos pontos em que nos focaremos no futuro, de maneira a melhorar os nossos resultados.

REFERÊNCIAS

ENDEREÇOS WEB

TUTORIAIS E BIBLIOTECAS

Daniel Shiffman. (2019). Coding Challenge #147: Chrome Dinosaur Game (with Speech Commands machine learning model!). Consultado em 8 mai. 2021. Disponível em: https://www.youtube.com/watch?v=l0HoJHc-63Q

Daniel Shiffman. (2016). 17.4: Amplitude Analysis - p5.js Sound Tutorial. Consultado em 10 mai. 2021. Disponível em: https://www.youtube.com/watch?v=NCCHQwNAN6Y

Daniel Shiffman. (2016). Flappy Bird Coding Challenge #31. Consultado em 12 mai. 2021. Disponível em: https: //thecodingtrain.com/CodingChallenges/031-flappybird.html

Jason Erdreich. (2020). p5js - Invaders pt5 - scoreboard. Consultado em 16 mai. 2021. Disponível em: https://www.youtube.com/watch?v=W_CmwdgBpG0

Maria Eduarda. (2020). Prova 1: Menu jogo p5.js. Consultado em 16 mai. 2021. Disponível em: https://www.youtube.com/watch?v=-879kZxnq7k

Mike “Pomax” Kamermans. (????). Let’s make a Mario game. Consultado em 16 mai. 2021. Disponível em: http://processingjs.nihongoresources.com/test/PjsGameEngine/docs/tutorial/mario.html

Daniel Shiffman. (2016). Q&A #1: Side-Scroller in p5.js. Consultado em 20 mai. 2021. Disponível em: https://www.youtube.com/watch?v=Ouza_4SsbLc

Daniel Shiffman. (2018). Coding Challenge #101: May the 4th Scrolling Text. Consultado em 9 jun. 2021. Disponível em: https://www.youtube.com/watch?v=fUkF-YLLXeg

Daniel Shiffman. (2019). Coding Challenge #131: Bouncing DVD Logo. Consultado em 10 jun. 2021. Disponível em: https: //www.youtube.com/watch?v=0j86zuqqTlQ

Ben Moren. (2020).p5.collide2D. Consultado em 10 jun. 2021. Disponível em: https: //github.com/bmoren/p5.collide2D

Daniel Shiffman. (2015). 7.8: Objects and Images - p5.js Tutorial. Consultado em 12 jun. 2021. Disponível em: https://www.youtube.com/watch?v=FVYGyaxG4To&list=PLnQ8GrlKNL7CL3gv0ClF_JgkgEd0TfhKm&index=5

Jacob Rivkin. (2020). p5.js - Random Image from Array. Consultado em 12 jun. 2021. Disponível em: https://www.youtube.com/watch?v=hxjEl-pun7o

EFEITOS SONOROS

soundboard nothing but sounds. (2021). Star Wars Sound Effects. Consultado em 18 mai. 2021. Disponível em: https://www.soundboard.com/sb/starwarsfx

mixkit. (2021). Free Game Sound Effects. Consultado em 18 mai. 2021. Disponível em: https://mixkit.co/free-sound-effects/game/

mixkit. (2021). Free Coin Sound Effects. Consultado em 18 mai. 2021. Disponível em: https://mixkit.co/free-sound-effects/coin/

MÚSICAS

8 Bit Universe. (2020). Star Wars Cantina Theme. Consultado em 17 mai. 2021. Disponível em: https://3mp3.buzz/mp3-download/8-bit-universe-star-wars-cantina-theme.html

8 Bit Universe. (2020). The Mandalorian Theme [8 Bit Tribute to Ludwig Göransson] - 8 Bit Universe. Consultado em 5 jun. 2021. Disponível em: https://www.youtube.com/watch?v=eoUsbydH-JKA&list=PLUO8mnyUG2EIfhwBE69y5H3A0BEZj-sah&index=13

8 Bit Universe. (2019). Star Wars Theme (2019 Remaster) [8 Bit Tribute to John Williams & Star Wars] - 8 Bit Universe. Consultado em 5 jun. 2021. Disponível em: https://www.youtube.com/watch?v=bOYdk1UY5o8

8 Bit Universe. (2014). Star Wars Imperial March Theme (8 Bit Remix Cover Version) - 8 Bit Universe. Consultado em 5 jun. 2021. Disponível em: https://www.youtube.com/watch?v=jRp-Pp0RCh-4&list=PLUO8mnyUG2EIfhwBE69y5H3A0BEZj-sah&index=3

IMAGENS

Img1. The Coding Train, Daniel Shiffman, 22/09/2017, https://www.youtube.com/watch?v=PBsU-D40nPkI

Img2. Star Wars: Tiny Death Star Achievements, exophase, https://www.exophase.com/game/star-wars-tiny-death-star-android/achievements/

Img3. RETROLYMPIAD, Ivan Dixon, 2016, https://www.ivandixon.com/retrolympiad

Img4. Printscreen do código, splice, 2021

Img5. Printscreen do código, hits, 2021

Img6. Printscreen do código, restart, 2021

Img7. Diagrama explicativo do código, 2021

Img8. Printscreen de gameplay de Star Wars Pamado Adventures, 2021

Img9. Printscreen de Miffy antes e depois, 2021

Img10. Printscreen de Pamado antes e depois, 2021

Img11. Printscreen de obstáculos, asteroides e nave, 2021

Img12. Free Arcade Machine Design Mockup in PSD, Designhooks, 2021, https://designhooks.com/freebies/free-arcade-machine-design-mockup-psd/

Img13. Free MacBook Pro mockup, MOCKUP-DESIGN.COM, 2021, https://mockups-design.com/free-macbook-pro-mockup/

Img14. Poster promocional desenvolvido, 2021

Img15. Free wrinkled poster mockup, MOCKUP-DESIGN.COM, 2021, https://mockups-design.com/free-wrinkled-poster-mockup/