ESCOLA SECUNDÁRIA DA MAIA
Curso Profissional de Gestão e Programação de Sistemas Informáticos
Ciclo de formação 2014/2017

PROVA DE APTIDÃO PROFISSIONAL

Survive Dungeon

Nazar Poritskiy

Julho 2017

Cofinanciado por:
POCH PORTUGAL 2020 UNIÃO EUROPEIA

ESCOLA SECUNDÁRIA DA MAIA
Curso Profissional de Gestão e Programação de Sistemas Informáticos
Ciclo de formação 2014/2017

PROVA DE APTIDÃO PROFISSIONAL

Survive Dungeon

Orientadores: **José Dias, Carla Malafaia, Manuel Jesus, Ruí Ribeiro**

Nazar Poritskiy

Julho 2017

Cofinanciado por:

Agradecimentos

A realização deste projeto foi possível e exequível graças ao conteúdo lecionado, e posteriormente desenvolvido por mim, por parte de todo o elenco docente nos quais passo a citar os nomes: Professora Carla Malafaia, Professor José Dias, Professor Manuel Jesus, Professor Rui Ribeiro.

Também grande obrigado a meu amigo e colega Luís Nunes por me ajudar com *design* do jogo.

A todo o elenco que indiretamente esteve envolvido, um sincero e enorme obrigado!

Descrição sumária

No âmbito da realização da Prova de Aptidão Profissional do curso “Técnico de Gestão e Programação de Sistemas Informáticos ”, desenvolvi um projeto com o objetivo criar um jogo para navegador de género aventura e com criação aleatória dos níveis, onde pretendi aplicar os conhecimentos adquiridos ao longo do curso e ainda adquirir novos.

A aplicação atual encontra-se em estado de desenvolvimento porque posteriormente com mais estudo e tempo vai ser mais desenvolvida tanto a nível gráfico como no conteúdo jogável e musical. Embora este facto não invalida o perfeito funcionamento, desenvolvimento, interação com o utilizador da mesma.

Abstract

In the scope of the Professional Aptitude Test of the course "Technician of Management and Programming of Computer Systems", I developed a project with the objective to create a game for navigator of adventure genre and with random levels creation, where I wanted to apply the knowledge acquired to the Course and still acquire new ones.

The current application is in a state of development because later with more study and time will be more developed both graphic level and playable and musical content. Although this fact does not invalidate the perfect functioning, development, software-user interaction of it.

Índice

1. Introdução 1
    1.1. Apresentação do projeto 1
    1.2. Objetivos 1
    1.3. Planeamento do projeto 2
    1.4. Contributos deste trabalho 2
2. Descrição técnica 2
    2.1. Análise 3
    2.2. Desenvolvimento 3
3. Resultados 12
4. Conclusões 15
    4.1. Objetivos concretizados 15
    4.2. Outros trabalhos realizados 16
    4.2. Limitações e trabalho futuro 16
    4.3. Apreciação final 17
5. Webgrafia 17
6. Anexos 18
    6.2. Animações 18
        6.1.1 Animação do Boss 18
        6.1.2 Animação do inimigo 18

Índice de figuras

Figura 1 Planeamento do projeto 2

Figura 2 Visualização da comunicação entre Utilizador e Base de dados 3

Figura 3 Código da criação do ambiente 3D 4

Figura 4 Código da criação dos objetos 4

Figura 5 Visualização dos objetos criados 5

Figura 6 Código da criação da luz 5

Figura 7 Código da criação dos *controlos* 6

Figura 8 Exemplo 1 do ambiente criado 6

Figura 9 Exemplo 2 do ambiente criado 7

Figura 10 Código do algoritmo de criação das paredes 8

Figura 11 Exemplo 1 de criação das paredes 8

Figura 12 Exemplo 2 de criação das paredes 9

Figura 13 Código da criação das paredes aleatórias 10

Figura 14 Exemplo 1 de criação aleatória das paredes 11

Figura 15 Exemplo 2 de criação aleatória das paredes 11

Figura 16 Código do carregamento das texturas 11

Figura 17 Código do carregamento dos modelos 3D 12

Figura 18 Código da criação dos inimigos 12

Figura 19 Código do adicionamento das músicas 13

Figura 20 Exemplo 1 do código do botão para desligar musica 13

Figura 21 Exemplo 2 do código do botão para desligar musica 13

Figura 22 O modelo 3D da chave no ambiente do jogo 14

Figura 23 O inimigo no ambiente do jogo 14

Figura 24 O menu/loja do jogo 15

Figura 25 A Tabela dos *leaders* 15

Figura 26 Janela do registo/entrada do jogo 16

Figura 27 Layout do nível normal 16

Notação e glossário

WebGL	Web Graphics Library
OpenGL	Open Graphics Library
HTML5	HyperText Markup Language v5
PHP	PHP: Hypertext Preprocessor
JS	JavaScript
Ajax	Asynchronous JavaScript And XML
MySQL	My Structured Query Language

1. Introdução

No âmbito da realização da Prova de Aptidão Profissional do curso “Técnico de Gestão e Programação de Sistemas Informáticos ”, desenvolvi um projeto com o objetivo criar um jogo para correr em ambiente de navegador *web*, onde pretendi aplicar os conhecimentos adquiridos ao longo dos três anos do curso e ainda adquirir novos.

1.1. Apresentação do projeto

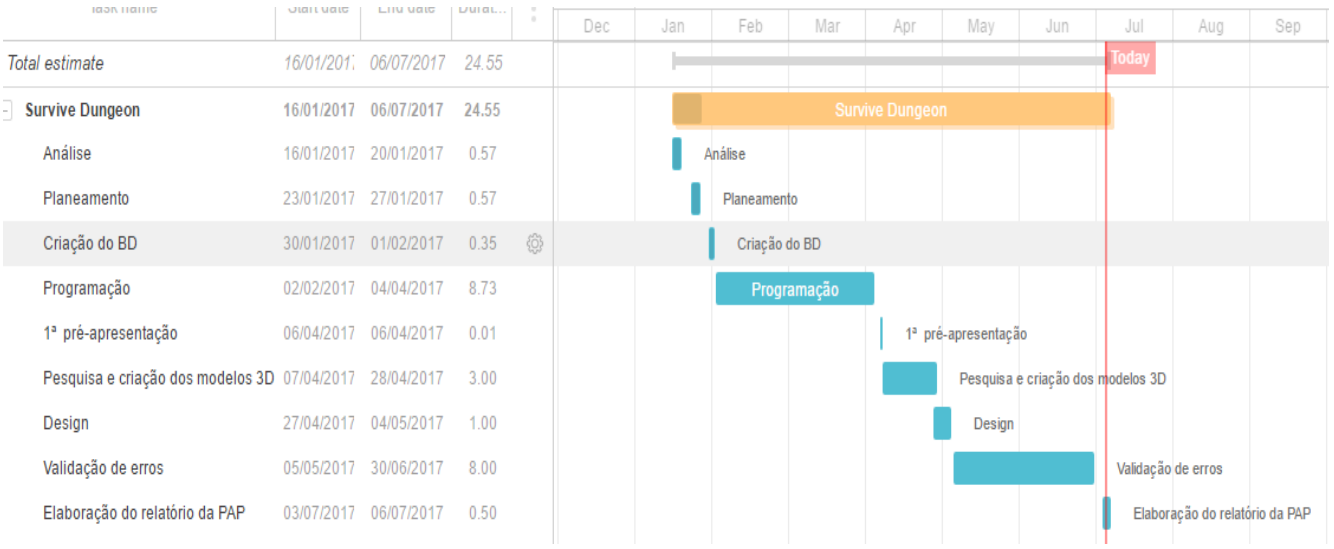
O tema do projeto era escolhida por mim e constitui na criação do jogo para web. O género do jogo é aventura, sobrevivência.

Depois de entrar no jogo o objetivo de cada nível é encontrar a chave e porta para acabar o nível. Cada nível é criado aleatoriamente e, depois de morte do jogador, o nível é criado de novo, pelo que não há níveis iguais. Cada próximo nível traz mais inimigos e um mapa maior. Ao longo do nível o jogador pode encontrar baús que aumentam o valor do dinheiro do jogador. O dinheiro serve para aumentar a velocidade ou a vida do jogador.

1.2. Objetivos

- Criar um jogo utilizando as linguagens de programação para *web*.
- Disponível para todos os computadores, independentemente das características de cada um.
- Cada nível tem que ser criado aleatoriamente.
- Entre os níveis normais colocar uns níveis com *Bosses* (adversários mais fortes, atribuindo uma maior dificuldade).
- Adicionar músicas nos níveis para manter concentração no jogo.
- Criar tabela dos *leaders* onde jogador pode se comparar com os outros utilizadores de jogo.

1.3. Planeamento do projeto



Total estimate	*16/01/201*	*06/07/2017*	*24.55*
Survive Dungeon	**16/01/2017**	**06/07/2017**	**24.55**
Análise	16/01/2017	20/01/2017	0.57
Planeamento	23/01/2017	27/01/2017	0.57
Criação do BD	30/01/2017	01/02/2017	0.35
Programação	02/02/2017	04/04/2017	8.73
1ª pré-apresentação	06/04/2017	06/04/2017	0.01
Pesquisa e criação dos modelos 3D	07/04/2017	28/04/2017	3.00
Design	27/04/2017	04/05/2017	1.00
Validação de erros	05/05/2017	30/06/2017	8.00
Elaboração do relatório da PAP	03/07/2017	06/07/2017	0.50

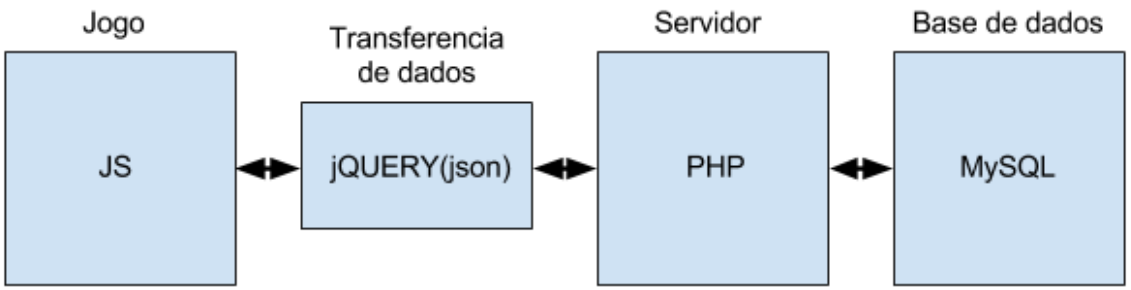
Figura 1 Planeamento do projeto

1.4. Contributos deste trabalho

Considero que o projeto “Survive Dungeon” surtiu aspetos de inovação perante todos os objetivos do curso, pois foi desenvolvido em JavaScript, uma das linguagens mais usadas em ambiente *web*, do que todas aquelas lecionadas e compreendidas durante o percurso escolar neste três anos de curso.

2. Descrição técnica

O jogo é escrito em JavaScript ou seja utiliza lógicas da linguagem. Para fazer o ambiente 3D no navegador utilizei WebGL, que é baseado em OpenGL. O WebGL usa o elemento *canvas* do HTML5 e a gestão automática da memória e é fornecida como parte da linguagem JavaScript .A comunicação entre o jogo e a base de dados é feita com PHP. Como o JS não pode comunicar com o PHP diretamente, precisei usar pedidos jQUERY (ajax) para transmitir dados entre JS e PHP.



2.1. Análise

O WebGL era difícil para mim por isso decidi usar uma outra biblioteca. Existem diversas bibliotecas para desenvolvimento com WebGL por exemplo: GLGE, C3DL, Copperlicht, SpiderGL, SceneJS, Processing.js, Three.js, Babylon.js , Turbulenz, XB PointStream e CubicVR.js. Li num *site* que as melhores para desenvolvimento de jogos são :Three.js e Babylon.js, pelo que decidi escolher entre essas duas. Comecei a testar a Babylon.js mas a biblioteca tinha pouca documentação e quase não tinha tutoriais, por isso decidi usar Three.js.

Para criar os modelos 3D utilizei Blender 3D. Escolhi esse programa porque é de código aberto, ou seja adicionando um *plugin* consegui exportar os modelos em formato .json e com facilidade usar no projeto.

2.2. Desenvolvimento

As primeiras partes do código eram muito simples. Comecei com criação da “scene”, adicionei a “câmera”. As duas imagens seguintes mostram o código e o resultado do mesmo.

```
scene= new THREE.Scene();

camera = new THREE.PerspectiveCamera(45,window.innerWidth /
window.innerHeight,0.1,5000);
camera.position.set(0,player.height,-5);
camera.lookAt(new THREE.Vector3(0,player.height,0));
```

Figura 3 Código da criação do ambiente 3D

O próximo passo era adicionar o chão e criar um objeto simples (cubo).

```
mesh = new THREE.Mesh(new THREE.CubeGeometry(1,1,1),new
THREE.MeshPhongMaterial({color:0xff4444,wireframe:false}));
mesh.position.y +=1;
mesh.receiveShadow = true;
mesh.castShadow = true;
scene.add(mesh);

meshfloor = new THREE.Mesh(new THREE.PlaneGeometry(100,100),new
THREE.MeshPhongMaterial({color:0xffffff,wireframe:false}));
meshfloor.receiveShadow = true;
scene.add(meshfloor);
meshfloor.rotation.x -= Math.PI/2;
```

Figura 4 Código da criação dos objetos

Figura 5 Visualização dos objetos criados

Depois adicionei os controlos para poder interagir com o ambiente e uma luz para iluminar melhor o cubo.

```
light = new THREE.PointLight(0xffffff,1,18);
light.position.set(-3,6,-3);
light.castShadow=true;
light.shadow.camera.near = 0.1;
light.shadow.camera.far = 25;
scene.add(light);
```

Figura 6 Código da criação da luz

```
if(keyboard[87]){ <!--w key-->
    camera.position.x -= Math.sin(camera.rotation.y) * player.speed;
    camera.position.z -= -Math.cos(camera.rotation.y) *
    player.speed;
}

if(keyboard[83]){ <!--s key-->
    camera.position.x += Math.sin(camera.rotation.y) * player.speed;
    camera.position.z += -Math.cos(camera.rotation.y) *
    player.speed;
}

if(keyboard[65]){ <!--a key-->
    camera.position.x += Math.sin(camera.rotation.y + Math.PI/2) *
    player.speed;
    camera.position.z += -Math.cos(camera.rotation.y + Math.PI/2) *
     player.speed;
}

if(keyboard[68]){ <!--d key-->
    camera.position.x += Math.sin(camera.rotation.y - Math.PI/2) *
    player.speed;
    camera.position.z += -Math.cos(camera.rotation.y - Math.PI/2) *
     player.speed;
}

if(keyboard[37]){ <!--left arrow-->
    camera.rotation.y -= player.turnspeed;
}
if(keyboard[39]){<!--right arrow-->
    camera.rotation.y += player.turnspeed;
}
```

Figura 7 Código da criação dos *controlos*

O código acima teve o seguinte resultado:

Figura 8 Exemplo 1 do ambiente criado

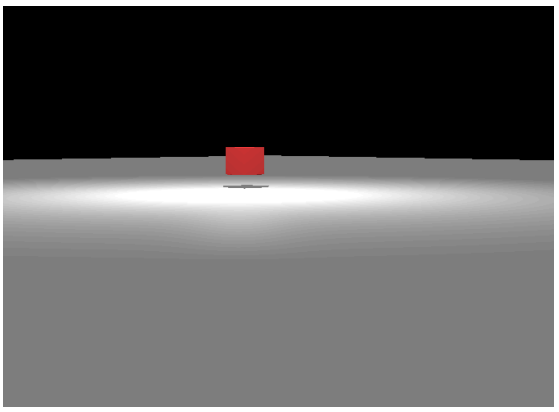


Figura 9 Exemplo 2 do ambiente criado

O próximo passo era adicionar um mapa que fosse criado aleatoriamente. Demorei algum tempo para desenvolver o algoritmo da criação do mapa, como na implementação dele em código. A ideia do algoritmo é criar paredes em duas etapas: primeiro ele cria paredes horizontais, de seguida as verticais.

```
var nelem=5;
var widthw=10;
var heightw=10;
var l=0;
var d=0;
for(var i=0;i<nelem;i++){
l+=widthw;
d=0;
for(var j=0;j<nelem+1;j++){
mesh1 = new THREE.Mesh(new THREE.CubeGeometry(widthw,heightw,1),new
 THREE.MeshNormalMaterial());
mesh1.position.z = d+(widthw/2);
mesh1.position.y = (widthw/2);
mesh1.position.x = l+(widthw/2);
scene.add(mesh1);
};
};
l=0;
d=0;
for(var i=0;i<nelem+1;i++){
l+=widthw;
d=0;
for(var j=0;j<nelem;j++){
mesh2 = new THREE.Mesh(new THREE.CubeGeometry(widthw,heightw,1),new
 THREE.MeshNormalMaterial());
mesh2.position.z = d+widthw;
mesh2.position.y = (widthw/2);
mesh2.position.x = l;
mesh2.rotation.y = Math.PI/2;
scene.add(mesh2);
d+=widthw;
};
};
```

Figura 10 Código do algoritmo de criação das paredes

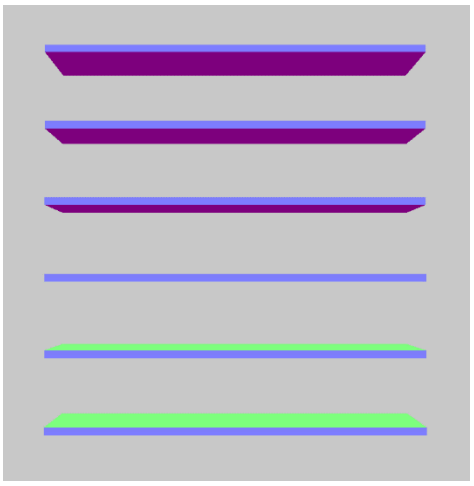


Figura 11 Exemplo 1 de criação das paredes

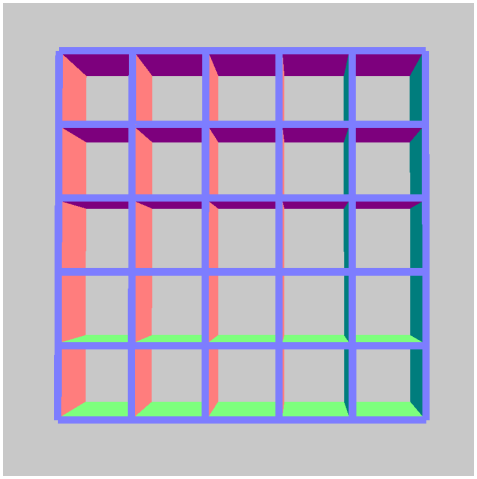


Figura 12 Exemplo 2 de criação das paredes

Depois adicionei a parte aleatória. A lógica é cada vez que o ciclo quer criar uma parede, ele cria um número aleatório de 0 a 7; se numero for 1 ou 5 ele cria a parede, se não deixa espaço vazio.

```
var nelem=5;
var widthw=10;
var heightw=10;
var l=0;
var d=0;
for(var i=0;i<nelem;i++){
l+=widthw;
d=0;
for(var j=0;j<nelem+1;j++){
var r = Math.floor(Math.random()*7);
if(r==1 || r==5){
mesh1 = new THREE.Mesh(new THREE.CubeGeometry(widthw,heightw,1),new
 THREE.MeshNormalMaterial());
mesh1.position.z = d+(widthw/2);
mesh1.position.y = (widthw/2);
mesh1.position.x = l+(widthw/2);
scene.add(mesh1);
d+=widthw;
}else{
d+=widthw;
};
};
};
l=0;
d=0;
for(var i=0;i<nelem+1;i++){
l+=widthw;
d=0;
for(var j=0;j<nelem;j++){
var g = Math.floor(Math.random()*7);
if( g==1 || g==5){
mesh2 = new THREE.Mesh(new THREE.CubeGeometry(widthw,heightw,1),new
 THREE.MeshNormalMaterial());
mesh2.position.z = d+widthw;
mesh2.position.y = (widthw/2);
mesh2.position.x = l;
mesh2.rotation.y = Math.PI/2;
scene.add(mesh2);
d+=widthw;
}else{
d+=widthw;
};
};
};
```

Figura 13 Código da criação das paredes aleatórias

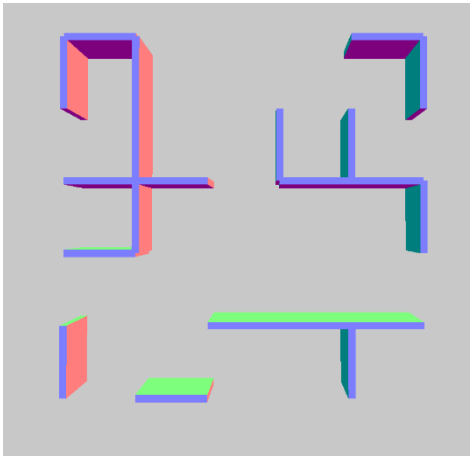


Figura 14 Exemplo 1 de criação aleatória das paredes

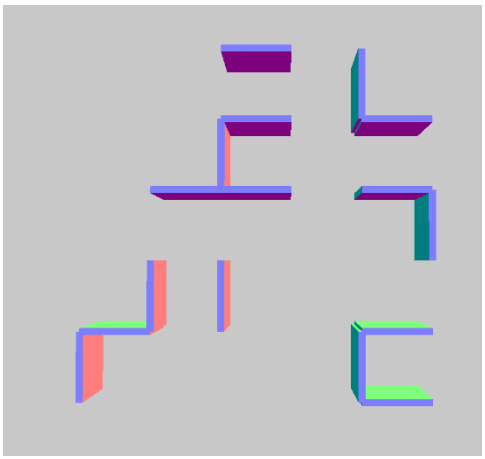


Figura 15 Exemplo 2 de criação aleatória das paredes

O próximo passo são as texturas do jogo, o carregamento das texturas é simples e repetitivo.

```
var textureLoader =new THREE.TextureLoader();
//wall texture
var wallTexture = new textureLoader.load("texture/stone_3.png");
wallTexture.wrapS = wallTexture.wrapT = THREE.RepeatWrapping;
wallTexture.repeat.set( 2, 2);
```

Figura 16 Código do carregamento das texturas

O carregamento dos modelos 3D é diferente. Por exemplo o modelo da chave é guardado num ficheiro. .json que guarda a geometria do modelo , quando o carregamento é feito o JS cria objeto com geometria da chave e posiciona no sítio aleatório do mapa .

```
var loader = new THREE.JSONLoader();
    loader.load('model/key2.json',function(geometry){
        mesh4 =new THREE.Mesh(geometry, new
        THREE.MeshLambertMaterial({color:0xad6903}));
        mesh4.name="key";
        objects.push(mesh4);
        var pkx=Math.random()*(widthw*(nelem+1))+5;
        var pkz=Math.random()*(widthw*(nelem+1))+5;
        mesh4.rotation.x += -Math.PI/2;
        if (lastboss==1){
            mesh4.position.set(boss.position.x,25,
            boss.position.z);
        }else{
            mesh4.position.set(pkx,7,pkz);
        }
        scene.add(mesh4);

    });
```

Figura 17 Código do carregamento dos modelos 3D

Depois de termos a base do jogo feito adicionamos os inimigos. Eles são criados num ciclo onde a variável “nenemy” corresponde ao número de inimigos e é calculada dividindo o número do nível ao meio. Como cada inimigo criado tem que ter a sua variável era utilizado o elemento “window” que deixa criar variáveis com JS a correr.

```
for(i=0;i<nenemy;i++){
    window['enemy'+i] = new THREE.Mesh(new THREE.CubeGeometry(11,
    0.1,11),new THREE.MeshBasicMaterial({color: 0xff0000, wireframe
    :false ,transparent: true,opacity:0}) );
    window['enemy_a'+i]= new THREE.Mesh(new THREE.PlaneGeometry(15,
    15),runnerMaterial);
    window['enemy'+i].castShadow = true;
    window['enemy'+i].receiveShadow = true;
    scene.add(window['enemy'+i]);
    scene.add(window['enemy_a'+i]);
    window['enemy'+i].name="enemy";
    objects.push(window['enemy'+i]);
    var p1x = Math.floor(Math.random()*((widthw*nelem)-10));
    var p1z = Math.floor(Math.random()*((widthw*nelem)-10));
    if(p1x<10)p1x+=5;
    if(p1z<10)p1z+=5;
    window['enemy'+i].position.set(p1x,0,p1z);
};
```

Figura 18 Código da criação dos inimigos

E por último a música que toca no nível. Código para adicionar música é simples mas a música em si é muito importante para manter o jogador concentrado.

```
audio = new Audio('audio/soundboss.m4a');
audio.play();
audio.loop=true;
audio.volume = 0.5;
```

Figura 19 Código do adicionamento das músicas

Se jogador não gostar da música ele sempre pode desligar la com um botão.

```
<button onclick="play()" class="menuButton" id="sound"><i id="mute" class="fa
fa-volume-up"></i></button>
```

Figura 20 Exemplo 1 do código do botão para desligar musica

```
function play(){
    if(audio.paused){
        audio.play();
        document.getElementById("mute").classList.remove(
        'fa-volume-off');
        document.getElementById("mute").classList.add('fa-volume-up');
    }else{
        audio.pause();
        document.getElementById("mute").classList.remove('fa-volume-up'
        );
        document.getElementById("mute").classList.add('fa-volume-off');
    }

}
```

Figura 21 Exemplo 2 do código do botão para desligar musica

3. **Resultados**

De seguida apresento alguns ecrãs do jogo.

Figura 22 O modelo 3D da chave no ambiente do jogo

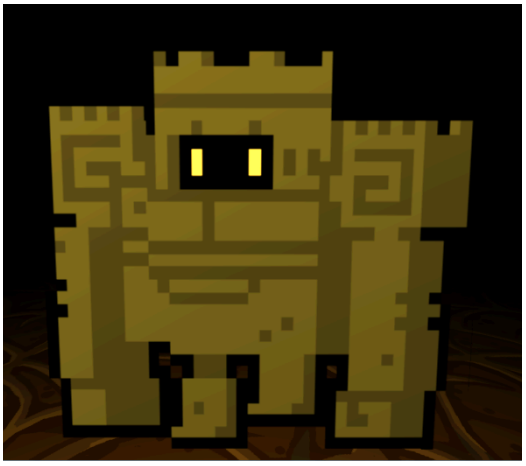


Figura 23 O inimigo no ambiente do jogo

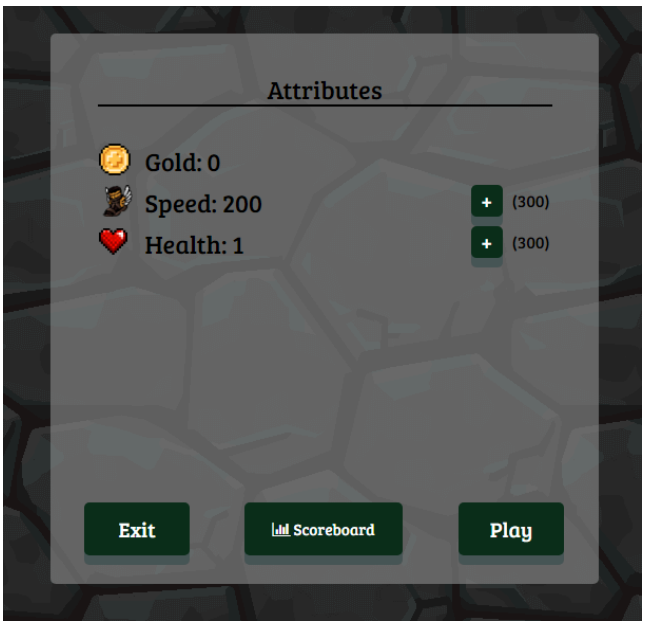


Figura 24 O menu/loja do jogo

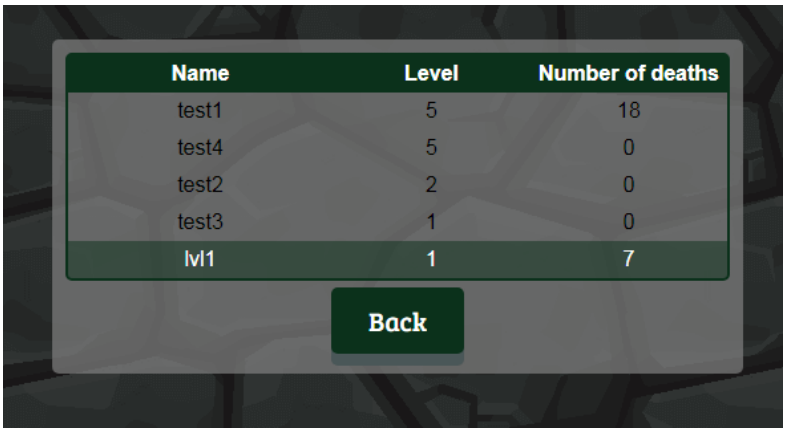


Figura 25 A Tabela dos *leaders*

Figura 26 Janela do registo/entrada do jogo

Figura 27 Layout do nível normal

4. Conclusões

Os objetivos previstos e estruturados neste projeto foram devidamente cumpridos com sucesso.

4.1. Objetivos concretizados

- Criar um jogo utilizando linguagens de programação para *web*.

O jogo foi criado e a está funcionar e preparado para apresentação.

- Disponível para todos os computadores, independentemente das características de cada um.
  Como ao longo do projeto quis fazer o jogo infinito por isso nos níveis mais elevados os jogadores com computadores mais fracos não podem jogar.
- Cada nível tem que ser criado aleatoriamente.
  Esses objetivo foi concretizado com sucesso.
- Entre os níveis normais colocar uns níveis com *Bosses* (adversários mais fortes, atribuindo uma maior dificuldade).
  Esses objetivo foi concretizado com sucesso. A cada cinco níveis aparece o "Boss".
- Adicionar músicas nos níveis para manter concentração no jogo.
  Esses objetivo foi concretizado com sucesso. Adicionei só duas músicas no nível normal e no nível com "Boss".
- Criar tabela dos *leaders* onde jogador pode comparar-se com os outros utilizadores de jogo.
  Esses objetivo foi concretizado com sucesso. Cada jogador pode ver a tabela com todos os jogadores e a estatísticas associadas.

4.2. Outros trabalhos realizados

O jogo foi colocado num domínio. Assim, o jogo está disponível *online*.

4.2. Limitações e trabalho futuro

Uma das limitações que me surgiu ao longo do desenvolvimento do projeto foi a modelagem 3D. Como nós nunca trabalhamos com o modelagem nas aulas, tive que estudar sozinho e, depois de algumas tentativas, percebi que é muito difícil pelo que decidi criar os inimigos e o *Boss* como *sprites* 2D.

O projeto pode continuar a ser desenvolvido de modo a atingir outras funcionalidades e disponibilizar ao jogador outras experiências de jogabilidade.

Trabalho para futuro:

- Adicionar os inimigos diferentes
- Mudar *sprites* 2D para aos modelos 3D

- Adicionar mais músicas

4.3. Apreciação final

Em suma, o jogo ”Survive Dungeon” com todas as suas atualizações até á data atingiu todas as expectativas do planeamento prévio do projeto, por isso, considero que cumpri todos os objetivos com sucesso e também os objetivos pessoais.

Perante tudo isto, pelo grau de complexidade, do que fui capaz de estudar, compreender e desenvolver considero que o jogo desenvolvido é um sucesso.

5. Webgrafia

https://threejs.org/	*Site* oficial da biblioteca onde estudei documentação toda
https://stemkoski.github.io/Three.js/	*Site* onde vi os exemplos
https://www.youtube.com/channel/UCVwgXgY_IVef4KqQh2liUxg	Canal Youtube onde vi os tutoriais

6. Anexos

6.2. Animações

6.1.1 Animação do *Boss*

6.1.2 Animação do inimigo