

# PANG



Trabalho feito por: Daniel Santos (40200100)

1.º Ano – Curso Técnico Superior Profissional de Design de Jogos e Animações Digitais

Disciplina: Programação e Animação Web

## 1. Introdução

Este trabalho foi feito para a disciplina de Programação e Animação Web e consiste no desenvolvimento do jogo clássico “Pang” usando HTML, CSS, Javascript e Phaser.

No Pang podem jogar 1 ou 2 jogadores, que tentam destruir uma bola usando um arpão cada um. Cada vez que uma bola é atingida o jogador do arpão que bateu na bola recebe pontos, e a bola divide-se em duas e fica com metade do tamanho anterior, até quando fica com um tamanho após 3 divisões e desaparece. Cada vez que uma bola atinge um jogador, esse jogador perde uma vida. Os jogadores começam com 3 vidas e passam de nível quando todas as bolas desaparecerem e ainda tiverem vidas. Se um jogador perder todas as suas vidas, perde o jogo. O jogo tem 4 níveis. Em cada nível também há um conjunto de “bricks” como no jogo do “Breakout”, em que as bolas batem e ao fim de um nº aleatório de batidas, cada tijolo desaparece e cai uma vida ou um asset que dá pontos ao jogador que os apanhar, para dar mais interesse ao jogo. Depois do jogo começar, qualquer jogador pode terminar imediatamente o jogo se quiser sair carregando na tecla ESC.





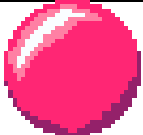


## 2. Repositório GIT e Código do Jogo

O repositório GIT tem os ficheiros Index.html e Javascript do código do jogo e também os ficheiros PNG e MP3 usados como assets dentro das pastas “images” e “sounds”.

### 2.1. Ficheiros JS

Ficheiro	Descrição
Index.html	Página inicial do jogo
Games.js	Configuração do jogo
Loadscene.js	Scene para carregar os assets do jogo
Mainscene.js	Scene principal do jogo
Player.js	Class sprite para um jogador
Brick.js	Class sprite para um brick (tijolo)
Ball.js	Class sprite para a bola
Arpoon.js	Class sprite para um arpão
Helth.js	Class sprite para uma vida
Points.js	Class sprite para ganhar pontos

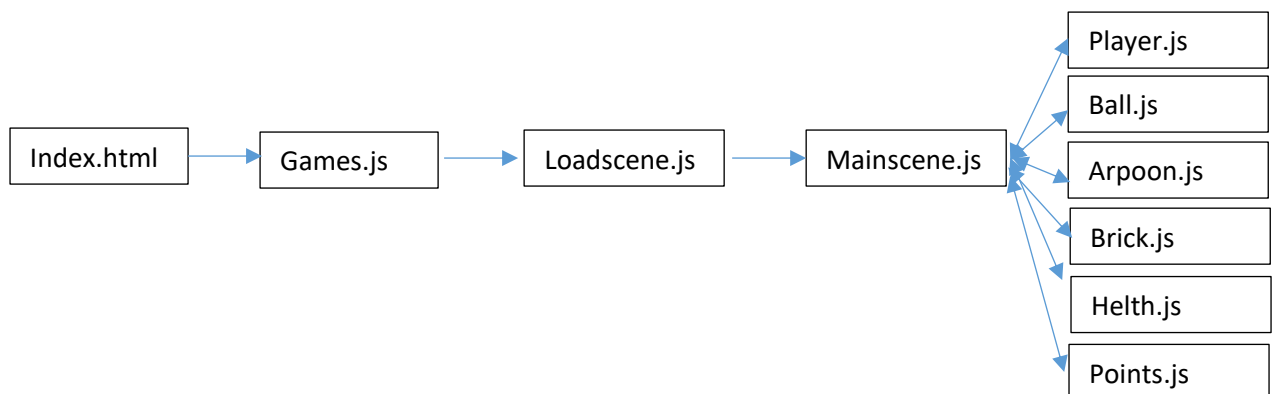
### 2.2. Ficheiros de assets de imagens e sprites

Ficheiro	Imagem/Sprite	Descrição
Background.png		Imagem de fundo do jogo
Peng_P1.png		Personagem do player 1
Peng_P2.png		Personagem do player 2
Peng_BoneArpoon.png		Arpão dos jogadores para atirar aos balões
Peng_Boll.png		Balão
Peng_Helth.png		Vida
Peng_Point.png		Pontos extra

### 2.3. Ficheiros de assets de audio

Ficheiro	Descrição
BK.mp3	Som de fundo do jogo
Hittingorjumpingball.mp3	Som quando uma bola destrói um tijolo ou é acertada por um arpão
Arpoon.mp3	Som de disparo de um arpão
Gotaliving.mp3	Som de ganhar uma vida por parte de um jogador
Gotpoint.mp3	Som de ganhar pontos extra por parte de um jogador
PlayerVoice1.mp3	Som quando um jogador é magoado por uma bola (usado aleatoriamente)
PlayerVoice3.mp3	Som quando um jogador é magoado por uma bola (usado aleatoriamente)
PlayerVoice4.mp3	Som quando um jogador é magoado por uma bola (usado aleatoriamente)
PlayerVoice5.mp3	Som quando um jogador é magoado por uma bola (usado aleatoriamente)
PlayerVoice6.mp3	Som quando um jogador é magoado por uma bola (usado aleatoriamente)

### 2.4. Diagrama de código



### 2.5. Classes, variáveis e funções

No Mainscene.js está basicamente toda a lógica do jogo, com as funções seguintes:

- constructor()
- create()
- keydown(event)
- createPlayers(players)
- repositionHarpoon(player)
- createAudioSources()
- update(time)

- `startGame()`
- `start_level(level)`
- `createMap(level)`
- `onCollisionBallPlayer(player, ball)`
- `playOuch()`
- `onCollisionBallArpoon(arpoon, ball)`
- `onCollisionPlayerHelth(player, life)`
- `onCollisionPlayerPoints(player, points)`
- `onCollisionBallBrick(ball, brick)`
- `updateHUD()`

## 2.6. Dificuldades e soluções

Fazer este jogo foi um bocado complicado.

Tive dificuldades em perceber como ter uma classe `Player` que pudesse colocar 2 jogadores ao mesmo tempo a movimentar-se com teclas diferentes. Para resolver isto criei uma função de `keyboard input` na `MainScene` e dois objetos `Player1` e `Player2` da classe `Player` e os seus arpões `Arpoon1` e `Arpoon2` da classe `Arpoon`, onde na função mexia cada um dos objetos dependendo das teclas.

Também tive dificuldades em como fazer para saber como ligar cada objeto da classe `Arpoon` ao seu `Player`, mas resolvi colocando uma variável `PlayerNumber` em cada `Arpoon` e depois uma função `setPlayer` na classe `Arpoon` para preencher essa variável com 1 ou 2 para saber qual o jogador a que pertence.

Uma das maiores dificuldades foi a falta de tempo por causa de outros trabalhos das outras disciplinas e principalmente o projeto interdisciplinar.

### 3. Gameplay

#### 3.1. Ecrã inicial

No ecrã inicial, são mostradas as teclas para escolher um ou dois jogadores, movimentá-los e começar a jogar.



#### 3.2. Jogando

Ao iniciar o jogo, são colocados os “bricks” espalhados na cena e aparece a bola de tamanho inicial que começa a mexer-se. O jogador começa a usar as teclas para se movimentar e atirar o arpão para acertar na bola. Existem 4 níveis e cada nível tem uma distribuição diferente de tijolos. O jogador passa de nível quando destruir todas as bolas desse nível e ainda tiver vidas.





Cada vez que o jogador acerta com o arpão na bola, ganha 100 pontos e ela divide-se em 2 bolas com metade do tamanho. Cada vez que uma bola acerta no jogador, ele emite um som e perde uma vida. As bolas dividem-se até 3 vezes antes de desaparecerem. Também os tijolos vão desaparecendo com um nº aleatório de batidas das bolas. Cada vez que desaparece um tijolo, cai uma vida ou pontos.



### 3.3. Terminar o jogo perdendo todas as vidas

Quando todos os jogadores já não tiverem vidas, o jogo termina.



### 3.4. Conseguir destruir todas as bolas até ao final do 4º nível – Ganhou!

Se o jogador conseguir destruir todas as bolas até ao final do 4º nível, chegou ao fim e ganhou o jogo.





## 4. Conclusão

Acho que conseguiu realizar o jogo proposto. Tive algumas dificuldades, mas consegui resolver os problemas e o jogo funciona, apesar de alguns pequenos “glitches”.

Acho que ficou um jogo divertido em que aprendi a usar o Phaser junto com HTML, CSS e Javascript para fazer jogos, apesar de eu gostar mais de design e desenvolvimento com ferramentas mais “user friendly” do que fazer código.