



## CRIAÇÃO DE UMA APLICAÇÃO PARA TELEMÓVEL PING PONG



**Aluno:** Telmo Miguel Ramos dos Santos

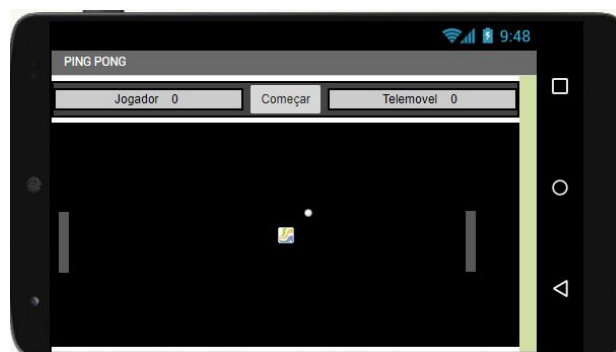
**Número:** a6707



Agrupamento de Escolas de Pinhel

Curso Profissional de Técnico de Multimédia

2018/2021



## CRIAÇÃO DE UMA APLICAÇÃO PARA TELEMÓVEL PING PONG

### PROVA DE APTIDÃO PROFISSIONAL

**Aluno:** Telmo Miguel Ramos dos Santos

**Número:** a6707

**Curso:** Técnico de Multimédia

**Diretor de Turma:** António Marques

**Coordenadora do Curso:** Ana Lourenço

## Agradecimentos

Antes de mais e em primeiro lugar, quero agradecer aos meus pais, por estarem sempre presentes, por acreditarem sempre em mim.

Agradeço também a todos os meus colegas de turma, por todos os momentos que vivemos juntos durante estes três anos.

A todos os professores por quem passei desde o décimo até ao décimo segundo ano.

Ao professor e Diretor de Turma, Paulo Monteiro, pelo apoio e a motivação extra que ele nos conseguia dar, e mesmo quando algo estava mal ele conseguia melhorar e resolver os problemas num estalar de dedos. Muito obrigado professor Paulo!

Agradeço também ao professor, António Marques, que nos acompanhou desde o início do curso, mas neste último ano de escolaridade, como director de turma, na minha perspetiva, o seu dever foi bem cumprido.

Agradeço também à coordenadora dos cursos e também professora de Português, Ana Lourenço, que fez com que nada nos faltasse durante estes três anos, um muito obrigado!

Por fim, mas não menos importante, agradeço ao senhor Diretor do Agrupamento, Dr. José Monteiro Vaz, por todo o apoio que nos deu, ao longo do curso e pelo voto de confiança.

## Resumo

De forma geral, a programação é um processo de escrita, testes e manutenção de programas de computadores. Esses programas, por sua vez, são compostos por conjuntos de instruções determinados pelo programador que descrevem tarefas a serem realizadas pela máquina e atendem diversas finalidades.

O meu gosto pela programação surgiu nos primeiros dias de aulas de Sistemas de Informação no 10º ano. Até ao dia a palavra programação, para mim, era praticamente desconhecida. No início até achei um pouco complicado, mas como se costuma dizer “quem corre por gosto não cansa”, e foi mais ou menos o que aconteceu comigo. Inicialmente achei difícil, porém com o passar do tempo comecei a ganhar gosto pela programação.

Surgiu a Prova de Aptidão Profissional, e como projeto decidi fazer algo que envolvesse programação, então criei um jogo para telemóvel e para tal vou usar o programa *AppInventor*.

## Palavras-Chave

Programação; AppInventor; Jogo

## Índice

Capítulo I – Introdução.....	6
1.1. Introdução .....	7
1.2. Escolha do Projeto .....	8
Capítulo II - Enquadramento Teórico .....	9
2.1. O Conceito de Programação .....	10
2.2. A História da Programação .....	11
2.3. Hardwares, softwares e programação .....	12
2.4. Linguagem de Programação .....	13
2.5. Código Fonte.....	13
2.6. Habilidades de um programador .....	13
2.7. A programação e o avanço das Tecnologias Digitais .....	15
2.8. O futuro da programação.....	16
2.10. Cuidados ao criar um jogo de telemóvel.....	19
2.11. História do videojogo de Ping Pong .....	23
Capítulo III – Programas para criar jogos para Telemóveis.....	24
3.1. AppInventor .....	26
3.2. Notepad++.....	27
3.3. SublimText.....	28
3.4. Microsoft Imagine .....	28
3.5. Codecademy.....	30
3.6. Scratch .....	31
3.7. Blockly .....	33
3.8. Cronapp.....	34
3.9. Codewars.....	35
3.10. Python .....	36
3.11. Mimo .....	37
3.12. EasyCoder .....	38
Capítulo IV – Desenvolvimento do Projeto .....	39
4.1. Aplicação utilizada.....	40
4.2. Barras de Ferramentas e Algumas Funcionalidades .....	41
4.3. Código da programação .....	42
4.4. Jogo Final .....	43

4.5. Divulgação do jogo .....	45
Capítulo V – Conclusões .....	46
5.1. Análise Crítica .....	47
5.2. Autoavaliação .....	48
5.3. Conclusão .....	49

## Índice de Figuras

Imagem 1 - Máquina Diferencial, ou Máquina Analítica – criada por Charles Babbage (1791 – 1871).....	10
Imagem 2 - Imagem Ilustrativa de Programação .....	25
Imagem 3 - Logotipo do App Inventor .....	26
Imagem 4 - Logotipo do Notepad++.....	27
Imagem 5 - Logoipo do SublimText .....	28
Imagem 6 - Logotipo Microsoft Imagine .....	29
Imagem 7 - Logotipo do Codecademy.....	30
Imagem 8 - Logotipo do Scratch.....	32
Imagem 9 – Logotipo do Blockly.....	33
Imagem 10 – Logotipo do Cronapp .....	34
Imagem 11 – Logotipo do Codewars .....	35
Imagem 12 – Logotipo do Python .....	36
Imagem 13 – Logotipo do Mimo .....	37
Imagem 14 – Logotipo do Easy Coder .....	38
Imagem 15 - Explicação das funcionalidades do AppInventor.....	41
Imagem 16 - Código do jogo.....	42
Imagem 17 - Jogo Finalizado .....	44
Imagem 18 – Alunos no Dia da Informática a jogarem no Tablet.....	45

## Capítulo I – Introdução



## 1.1. Introdução

A Prova de Aptidão Profissional faz parte integrante do curso e deve ser um projeto transdisciplinar integrador de todos os saberes e capacidades que foram adquiridas ao longo dos 3 anos.

Atualmente, a tecnologia desenvolvida nesta área, permite criar aplicações muito facilmente, uma vez que existem muitos programas para desenvolver jogos.

O meu projeto tem como objetivo criar um jogo de Ping Pong para telemóvel no programa *AppInventor*, tem uma questão sobre a cidade de Pinhel, para dar a conhecer um pouco a cidade.

O Ping Pong, também conhecido como ténis de mesa, é um dos desportos mais populares e mais praticados entre os jovens de todo o mundo. Se se tiver preguiça de ir ao clube, pode-se jogar Ping Pong virtualmente.

A elaboração deste projeto requereu um trabalho constante ao longo de todo o ano, que se iniciou com uma pesquisa aprofundada sobre o tema, a qual faz parte deste documento.

## 1.2. Escolha do Projeto

Ao longo do curso foram exploradas várias ferramentas nas diferentes áreas da multimédia, no entanto a que me despertou maior interesse foi sem dúvida a programação.

Neste sentido, a minha ideia de PAP surgiu quando, na disciplina de PPM, estávamos a iniciar a criação de aplicações para Android. Pensei então em criar um jogo de Ping Pong.

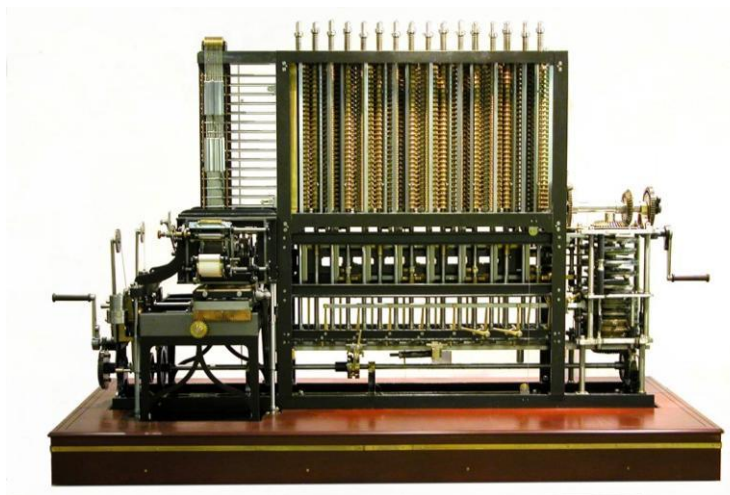
Assim, a vontade de explorar este tipo de ferramentas, aprofundando os meus conhecimentos numa área que está cada vez mais presente na sociedade, juntamente com o apoio dos meus professores e colegas levaram à concretização deste projeto.

## Capítulo II - Enquadramento Teórico

## 2.1. O Conceito de Programação

De forma geral, a programação é um processo de escrita, testes e manutenção de programas de computadores. Esses programas, por sua vez, são compostos por conjuntos de instruções determinados pelo programador que descrevem tarefas a serem realizadas pela máquina e atendem diversas finalidades.

O advento da programação está diretamente ligado à história dos computadores. A primeira programadora foi uma mulher chamada Ada Lovelace, que escreveu um código para possibilitar a utilização da máquina analítica de Charles Babbage, uma máquina robusta, de difícil comunicação, considerada a precursora dos computadores eletrônicos atuais.



**Imagem 1 - Máquina Diferencial, ou Máquina Analítica – criada por Charles Babbage (1791 – 1871)**

Um século depois, os computadores ficaram cada vez mais fáceis de serem utilizados e a programação ganha cada vez mais espaço na criação dessas tecnologias. Mas, para compreender de verdade do que ela se trata, é importante conhecer dois elementos fundamentais que compõem todos os computadores modernos, que são o *hardware* e o *software*.

## 2.2. A História da Programação

O primeiro trabalho de linguagem de programação foi criado por Ada Lovelace, grande amiga de Charles Babbage. O projeto da primeira calculadora mecânica programável foi idealizado por Charles Babbage que, após gastar fortunas e um longo tempo, não conseguiu concretizar o projeto. A linguagem de programação ADA foi batizada em homenagem a esta primeira programadora.

Uma das primeiras linguagens de programação para computadores foi provavelmente Plankalkül, criada por Konrad Zuse na Alemanha Nazista, mas que teve pouco ou nenhum impacto no futuro das linguagens de programação.

O primeiro compilador foi escrito por Grace Hopper, em 1952, para a linguagem de programação A-0. A primeira linguagem de programação de alto nível amplamente usada foi Fortran, criada em 1954. Em 1957 foi criada B-0, sucessora da A-0, que daria origem a Flow-Matic (1958), antecessor imediato de COBOL, de 1959. O COBOL foi uma linguagem de ampla aceitação para uso comercial. A linguagem ALGOL foi criada em 1958-1960. O ALGOL-60 teve grande influência no projeto de muitas linguagens posteriores.

A linguagem Lisp foi criada em 1958 e se tornou amplamente utilizada na pesquisa na área de ciência da computação mais proeminentemente na área de Inteligência Artificial. Outra linguagem relacionada ao campo da IA que surge em 1972 é a linguagem Prolog, uma linguagem do paradigma lógico.

A orientação a objetos é outro marco importante na história das linguagens de programação. A linguagem Simula 67 introduz o conceito de classes. A linguagem Smalltalk expande o conceito de classes e se torna a primeira linguagem de programação que oferecia suporte completo à programação orientada a objetos. A linguagem C++ (originalmente conhecida como C com classes) populariza a orientação a objetos.

## 2.3. Hardwares, softwares e programação

Um computador é formado por componentes físicos, como a parte externa do seu celular, a CPU de um PC, a memória, o rato, enfim... os meios pelos quais os sinais elétricos podem ser convertidos em dados, informações.

Esses componentes são chamados de *hardware* e a sua linguagem binária, que é a linguagem de máquina, é composta apenas por bits, que são zeros e uns. Os bits representam a ausência ou presença de sinais elétricos. Agora, imagine utilizar o seu telemóvel tendo que interpretar informações binárias: se tudo o que se vê na tela fosse uma sequência infinita de zeros e uns, os telemóveis não seriam nada práticos, não é mesmo? É por causa disso que existem os softwares.

O software é o meio pelo qual a linguagem de máquina pode ser compilada ou interpretada, através de códigos criados em uma linguagem intermediária, para idiomas que conhecemos, como o português, o inglês e também para imagens, cores, números... enfim, uma série de dados que podem ser assimilados mais facilmente pelos seres humanos.

Alguns exemplos de softwares são os programas de computadores, como o Word, Excel e Paint, e as aplicações para telemóveis, como a Playstore, App Store ou o Uber. Os sistemas operativos, como o Android, Windows, Linux ou o Mac, também podem ser vistos, de forma simplificada, como softwares gigantes que comportam outros softwares.

Mas, afinal, qual é a relação entre hardware, software e programação? A programação é exatamente quem possibilita a existência dos softwares e, por consequência, a utilização mais prática dos hardwares. Para poder dar origem aos softwares, a programação ganha uma linguagem própria que compõe códigos escritos por programadores.

## 2.4. Linguagem de Programação

A linguagem de programação nada mais é do que um método padronizado que permite comunicar instruções específicas para um computador/telemóvel. Há centenas de variações e todas elas permitem que o programador defina sobre quais os dados que o computador deverá atuar, como armazenar e transmitir esses dados e quais ações tomar sob variadas circunstâncias.

## 2.5. Código Fonte

Código fonte é o conjunto de palavras e comandos escritos ordenadamente, de maneira lógica, que contém instruções em determinada linguagem de programação.

Há dois tipos de linguagem na hora de programar: as compiladas e as interpretadas.

Compilar uma linguagem de programação é transformar o código fonte em algo que faça sentido para a máquina, ou seja, transformar o que o programador escreveu em linguagem de máquina.

Já no processo de interpretação, o código fonte é transformado numa linguagem intermediária que, por sua vez, será interpretada pela máquina virtual durante a execução do programa.

## 2.6. Habilidades de um programador

É importante também saber que muito além de um simples processo de escrita, a programação envolve um misto de outras habilidades cognitivas, como o raciocínio

lógico, habilidades matemáticas, a capacidade de abstração e de lidar com diferentes tipos de dados.

Quando se começa a estudar programação, a primeira coisa que se aprende é criar algoritmos básicos, que são como receitas, para que a partir de um passo a passo, uma tarefa possa ser executada. E esse algoritmo é desenvolvido a partir de tentativas e erros baseados no resultado em que se quer chegar e em como se pode alcançá-lo.

Um exemplo disso é quando pede para uma pessoa ir a tal padaria, ela vai precisar saber onde ela se localiza para passar as instruções adequadas, como: andar em frente até à rua tal (e também precisará saber o nome da rua) vire à direita (e precisará ter noções de direção) e assim por diante.

Com isso, estará usando a lógica básica para criar um mapa de instruções, assim como os programadores começam a exercitar a lógica para chegar a softwares de computadores, tendo em vista que eles estão instruindo a máquina a chegar a um resultado.

Seguindo no exemplo acima, para chegar a caminhos mais complexos, como construir um prédio, ou criar um software para engenheiros, é preciso passar por esse caminho do básico e ir somando conhecimentos mais complexos, como conhecimentos matemáticos e da própria engenharia, por exemplo.

Por esse motivo é importante dedicar tempo a aprender programação, e quanto mais cedo se começa a dedicar à programação, mais cedo também se alcança resultados mais complexos, se tornando um programador cognitivamente capacitado para criar estratégias e soluções tecnológicas muito úteis à nossa sociedade, como softwares médicos, de engenharia, entre outros.



## 2.7. A programação e o avanço das Tecnologias Digitais

A programação teve o seu início há mais de um século, partindo das ideias da Ada Lovelace com o computador robusto do Babbage. Até ao período da segunda Guerra mundial, os hardwares ainda tinham um protagonismo muito maior nos estudos e pesquisas dessas tecnologias do que os softwares, que são o objeto de criação da programação.

Mas, depois da segunda guerra mundial, e principalmente quando os computadores passaram a ganhar uso além do bélico, como os primeiros computadores pessoais e, algumas décadas depois, com a criação dos smartphones, os aparelhos eletrónicos têm solicitado cada vez mais atenção aos softwares, que devem ser tão atualizados quanto os hardwares para que a tecnologia tenha visibilidade e atenda aos pedidos, cada vez mais exigentes, do mercado.

Além do uso pessoal, atualmente a programação tem grande importância também em áreas fundamentais à sociedade, como na medicina, onde muitos aparelhos médicos utilizam a monitoração remota e em tempo real, possibilitada pela programação, na engenharia, indústria mecânica, automotiva e arquitetura, onde vários softwares utilizados para a criação de projetos foram desenvolvidos por programadores, entre outras áreas.

Foi o avanço da programação que possibilitou também o que hoje chamamos de Internet das Coisas, que é capaz de automatizar desde pequenos ambientes de uma casa até uma cidade inteira.

Além disso, a programação é uma das principais ferramentas utilizadas no avanço das tecnologias artificiais, que já trazem (a exemplo da Cortana e de invenções como a robô Sophia) muito mais comodidade ao dia a dia, por exemplo ao fornecer informações em tempo real e ser capaz de realizar diversas atividades, como a Sophia, que atua em tratamentos biológicos como o autismo.

## 2.8. O futuro da programação

O novo e mais complexo sentido da palavra “Programação” tem adquirido uma relevância cada vez mais alta nas últimas décadas e, graças ao advento da Indústria 4.0, que é o contexto tecnológico em que estamos inseridos, esse cenário tende a ser permanente e exigir programadores cada vez mais capacitados para atuar numa série de diferentes áreas voltadas para a programação nos próximos anos.

A programação pode ser vista, assim, como uma profissão fundamental no futuro, e é justamente por isso que diversos países têm-se esforçado para iniciar o ensino de programação cada vez mais cedo para as crianças, para desenvolver neles habilidades de raciocínio lógico, abstração, entre outras e também para suprir as exigências crescentes da profissão.

## 2.9. Conceito de Telemóvel e sua Evolução

Telemóvel é um aparelho de comunicação por ondas eletromagnéticas que permite a transmissão bidirecional de voz e dados utilizáveis em uma área geográfica que se encontra dividida em células, cada uma delas servida por um transmissor/receptor. A invenção do telemóvel ocorreu em 1947 pelo laboratório Bell, nos Estados Unidos.

De facto, a indústria de telecomunicações tem designado saltos tecnológicos nos sistemas de comunicações móveis por gerações, que têm surgido com regularidade no início de cada década.

No início da década de 1980 começaram a aparecer vários sistemas em vários países, a 1ª geração (1G), tendo-se chegado a cerca de uma dezena de sistemas diferentes. Nestes sistemas, de tecnologia analógica, o único serviço disponível era o da voz, ou

seja, fazer chamadas telefónicas. Os telefones eram autênticos “caixotes” (basicamente, um volume de 5 litros e um peso de 5 quilos). Em Portugal, o sistema (de origem alemã, Rede-C) foi disponibilizado já no final da década.

Entretanto, a meio da década de 1980, devido ao movimento na Europa para a livre circulação de pessoas, foi colocado o desafio à indústria para desenvolver um sistema que permitisse às pessoas usar o seu telemóvel para comunicar noutro país (o que não era possível com o 1G), isto é, aquilo que hoje em dia conhecemos por roaming. Nascia assim a 2ª geração (2G) no início da década de 1990, o GSM, com tecnologia digital (o que, aliás, é comum a todos os sistemas posteriores), com telefones que inicialmente já eram só um “tijolo” e permitiam ainda só as chamadas telefónicas. O sucesso deste sistema ultrapassou todas as expectativas em termos de número de utilizadores: em Portugal, o 1G atingiu pouco mais de 30 mil utilizadores, enquanto o 2G tinha quase 5 milhões de utilizadores no início do ano 2000. O sistema conheceu várias atualizações que permitiram a utilização de outros serviços, nomeadamente o SMS (que teve um enorme sucesso) e a transmissão de dados, embora a velocidades que hoje em dia consideraríamos “ridículas” (inicialmente, era de 9,8 kbit/s, tendo atingido um máximo de cerca de 150 kbit/s).

A 3ª geração (3G), o UMTS, já foi desenvolvida num panorama completamente diferente do GSM, apoiando-se no sucesso deste e tendo surgido no início da década de 2000. Este sistema já permitia inicialmente velocidades de dados de 384 kbit/s, para além da voz e do SMS. O objetivo deste sistema era o de possibilitar as videochamadas, ou seja, fazer algo que também hoje em dia tomamos por banal, que é falar e ver alguém em simultâneo. O acesso à internet também já começava a banalizar-se, o que motivou o desenvolvimento de novas versões do sistema, que chegou a uma velocidade máxima atual de cerca de 40 Mbit/s. Os telefones atingiram o seu tamanho mínimo, mais pequeno que um maço de cigarros, e “quase” com um peso semelhante; o aparecimento do iPhone, em 2007, fez desaparecer o teclado físico do telefone, que passou a ser virtual no ecrã. A meio da década, em 2005, atingiram-se os 100% de utilizadores, ou seja, em média, cada português tinha um

telemóvel, o que significava que muitos utilizadores possuíam mais de um telefone (porque, de facto, nem toda a população o possuía).

No início da década de 2010, fomos apresentados à 4ª geração (4G), o LTE; com velocidades iniciais de 100 Mbit/s, que chegavam aos 300 Mbit/s. O acesso à internet (com todos os serviços que conhecemos) a partir dos telemóveis (agora designados por smartphones) e a utilização das apps são algo que se faz com uma absoluta normalidade, mas que motivou que o tamanho dos telefones começasse a crescer, para possibilitar uma melhor leitura da informação. Passou-se também à situação em que muitos utilizadores possuem um telefone, um tablet e um computador com acesso LTE.

A 5ª geração (5G) foi disponibilizada em 2020. Estes sistemas de comunicações móveis têm essencialmente duas componentes na sua arquitetura: a interface rádio e a rede de nós. A rede é constituída basicamente por equipamentos (na sua maioria computadores) ligados entre si por cabos de fibra ótica (nalguns casos pode haver ligação por feixe rádio), que efetuam o processamento e encaminhamento da informação. A interface rádio é aquela que permite a ligação entre o telemóvel e a estação base (equipamento que tem como aspeto mais visível as antenas colocadas em cima de telhados de edifícios ou no topo de torres).

Muitos dos novos desenvolvimentos ocorrerão ao nível da rede, ou seja, não serão diretamente perceptíveis pelos utilizadores, embora as suas consequências o sejam. Por outras palavras, para que o utilizador tenha acesso a comunicações “mais evoluídas”, a maneira como a informação é processada e transportada tem que ser modificada, devido a limitações na atual 4ª Geração. Essa evolução passa pela transmissão de dados com velocidade maior e atraso menor, e pela inclusão de dezenas de dispositivos na sua rede pessoal, entre outras.

## 2.10. Cuidados ao criar um jogo de telemóvel

Atualmente quase todos jogos de videogame tendem a ter imagens de qualidade cinematográfica, com computação gráfica avançada, e com gráficos em 3D. Portanto, todos esses atributos num jogo de telemóvel devem ser muito bem estudados antes de serem usados. O fator da tela do telemóvel também ter poucos pixels pode ser impeditivo para alguns desses recursos visuais. O ideal seria não usá-los, pela dificuldade de programação que isso geraria, pelo tamanho dos arquivos que criariam, e pela pouca fidelidade ao que existe hoje nos videogames. Querendo ou não, os jogadores vão se lembrar de algum jogo semelhante existente nas consolas atuais, e vão notar que não ficou fiel. Portanto, se há o risco de falhar um desenvolvimento de algo neste género, é recomendado repensar cuidadosamente a ideia.

Outro recurso que muitos jogos atuais têm é a conectividade e a possibilidade de jogadores múltiplos. Isto sim deve ser utilizado. Essa necessidade de múltiplos jogadores vem sendo notada, primeiramente pelo fato das consolas, que antigamente suportavam apenas dois controles sem adaptadores, hoje em dia apresentarem entradas para até quatro controles diretamente na consola; em segundo pelo grande número de títulos de jogos para múltiplos jogadores on-line (os MMORPG), que atualmente são vedetas do mercado de entretenimento digital. Investir nas tecnologias de comunicação com telemóveis pode ser uma forma de atrair diversos jogadores, que se divertirão competindo uns com os outros.

No entanto, os telemóveis têm baterias, e estas podem esgotar-se. Logicamente, existem formas de guardar o andamento do jogo, mas se um jogo pode acabar com uma carga de bateria, é porque provavelmente ele tem uma extensão muito longa para ser jogado em telemóvel. A menos que se tenha uma boa justificação para isso, são de evitar jogos muito longos. Antigamente, nos jogos para as primeiras consolas, não havia possibilidades de guardar o andamento, porém mesmo assim eles eram divertidos. Como nem todos os teclados de telemóveis são amigáveis para jogos, são ideais jogos que exijam mais da inteligência do que da rapidez dos dedos do jogador.

---

Assim, quebra-cabeças para resolver problemas no decorrer do jogo são recomendados. Sokoban é um ótimo exemplo de jogo que não tem ação, somente uso da mente. Para uma fila de espera indesejada, é uma ótima opção. Outros mais recentes, como Legend of Zelda, seguem uma amálgama entre os conceitos: uma boa dose de ação, misturada com inúmeros quebra-cabeças para serem resolvidos.

### **Enredos**

Para alguns formatos de jogos, como os de plataforma e aventuras, deve-se ter o cuidado de criar um bom enredo, uma história que segue as ações do personagem. Como telemóveis devem ter jogos enxutos em termos de tamanho em bytes e extensão de jogo, a premissa deles devem ser simples o bastante para ser explicada em no máximo duas linha de texto. Se não for, é necessário repensar o seu enredo. Ele pode ser complicado demais, difícil de se desenvolver e talvez não traga a diversão esperada.

Outra coisa que devemos ter cuidado é com a abordagem que terá o jogo. Ele será sério? Será cómico? Terá terror? Todos esses elementos devem ser definidos, pois uma miscelânea deles num único jogo pode tornar o jogo confuso e sem diversão. Definir uma única abordagem poderá facilitar para a criação do roteiro, dos personagens e da premissa.

Assim como a premissa, os personagens devem ser simples, para talvez um dia se tornarem clássicos. Pode-se também dar algum toque para que ele seja ímpar: algo como uma arma diferente, um problema pessoal ou algum traço de personalidade. E também é importante que eles sigam o mesmo contexto dos heróis. Caso não, quando um jogador chegar ao fim do jogo, poderá achar que o jogo não foi tão divertido quanto ele esperava. O vilão final deve impor uma dificuldade a mais, e de preferência uma dificuldade mental, para que ele seja vencido. Vilões sempre são tidos como mentores de problemas para os heróis, por isso é esperado que eles sejam mais inteligentes que o próprio jogador que controla o personagem. Vale a pena perder um bom tempo com mais linhas de código para fazer com que seu vilão seja mais difícil de ser vencido. Logicamente, deve-se assegurar que ele é possível de ser vencido... Sub-

chefes devem ser mais fáceis. Sempre são “sacos-de-pancada”, mas por serem chefes, lembrem-se que eles devem ter um motivo para estarem junto ao vilão e contra os heróis. Pode parecer forçado vencer um desafio que simplesmente estava ali para ser vencido. Outros inimigos mais simples não precisam de muito estudo por parte do desenvolvedor, mas devem entrar no contexto do jogo. Imagine este exemplo de como não se fazer inimigos simples: “Numa fase do jogo encontra-se uma mosca venenosa que ataca o herói, seguida por uma aranha gigante”. Esse exemplo trás a pergunta: “por que a aranha não comeu a mosca?”. E a exemplo dos sub-chefes, dar um motivo daquele quebra-cabeça existir naquele momento do jogo. Até agora temos formas de criar um enredo, heróis, vilões e demais personagens. Mas, para transmitir toda essa ideia para o jogador, como devemos seguir? Existem muitos exemplos. Um que realmente vale ser citado é a primeira trilogia de Ninja Gaiden (ou Ryukenden para alguns). Este jogo implantou uma forma de contar a história do jogo através de cenas como sequências de filmes. A sequência de apresentação do jogo era muito simples. Dois ninjas confrontam-se, e um deles morre. Aparecia, depois, o protagonista do jogo, Ryu Hayabusa, encontrando uma carta de seu pai aonde ele insinuava que iria morrer, e citava um nome de um homem. Seu pai era o ninja morto na apresentação, e com a sua morte Ryu resolve ir ao encontro do homem citado na carta, para saber o que levou o seu pai aquele duelo, e à sua morte. Entre uma fase e outra, seguiam cenas (em formato wide-screen, já se adiantando a tecnologia, mesmo sem ter a qualidade visual de um filme), onde geralmente os personagens apareciam com a legenda das suas falas, e com isso eles contavam os mistérios que o personagem principal ia descobrindo. Tudo muito bem encaixado, e justificando cada fase, cada local por onde o protagonista passava.

Vendo esses exemplos, notamos que a forma com a qual se vai envolver o jogador no enredo depende muito da sua afinidade em contar histórias. Uma boa dica para aprimorar essa técnica é ler romances, contos, quadrinhos e assistir a filmes. Observar como eles colocam o leitor/espectador dentro do enredo, dentro da história.

Analisando jogos de plataformas notamos que vale muito mais ideias diferentes e simples, do que inovadoras e complicadas.

Jogos de corrida e de desporto não necessitam de tanto roteiro. Basta que sejam divertidos. Para isso, não adianta trazer todas as variáveis da realidade para a tela do jogo. O ideal é que o jogo tenha facilidade em ser jogador, com comandos fáceis e rápidos, e que ele seja desafiador, colocando o jogador num nível que exija dele um esforço gradativo de acordo com a sua performance. Como um exemplo, existia um jogo chamado Super Sprint. Era um jogo estilo fórmula 1, com quatro carros que mais pareciam garrafas de refrigerantes com rodas. Tinha-se a visão superior de toda a pista, e os comandos eram acelerar, travar, direita e esquerda. As pistas eram muito divertidas, com atalhos e pontes. O jogo baseava-se em cada corredor cometer menos erros na pista, e não bater em seus adversários, pois quanto mais erros, mais fácil para os adversários ultrapassarem.

As teclas dos telemóveis, são o maior facto impeditivo para jogos de ação e de luta. Jogos estilo Street Fighter são totalmente fora de questão. Eles envolvem sequências de comandos que seriam impraticáveis nas teclas de telemóvel, sem contar a quantidade de botões distintos usados no jogo. A disposição das teclas em telemóveis é desprivilegiada para jogos, por isso é sempre bom o desenvolvedor ter em mente que muitos poderão ter uma tendinite tentando jogar o jogo que ele criou.



## 2.11. História do videojogo de Ping Pong

O primeiro videojogo de Ping Pong surgiu com o nome de Pong foi o primeiro videojogo lucrativo da história, dando origem a um novo setor da indústria. Foi de importância fundamental na história do videojogo. Foi criado por Nolan Bushnell e Ted Dabney na forma de uma consola ligado a um monitor, movido a moedas. A primeira instalação num bar de San Francisco, Califórnia, mostrou aos dois a possibilidade de lucro da criação. Assim, em 27 de Junho de 1972, a empresa Atari foi fundada.

O sucesso de *Pong* fez com que Bushnell criasse um novo produto. Em 1974, o engenheiro da Atari Harold Lee propôs uma versão doméstica de *Pong* que se conectaria à televisão: *Home Pong*. O sistema começou a ser desenvolvido sob o codinome *Darlene*, em homenagem a uma atrativa empregada da Atari.

*Home Pong* foi um sucesso instantâneo após o seu lançamento limitado em 1975 pela Sears; cerca de 150.000 unidades foram vendidas nas férias daquele ano. O jogo tornou-se o produto de mais sucesso de Sears na época, o que deu a Atari o Prémio de excelência em qualidade Sears.

## Capítulo III – Programas para criar jogos para Telemóveis

Como para tudo na vida, existem várias opções, em termos de programação não é exceção, pois o que não falta, são aplicações que nos ajudem a desempenhar as nossas funções, ou seja, neste caso criar ou reprogramar uma aplicação.



Imagem 2 - Imagem Ilustrativa de Programação

### 3.1. AppInventor

Este é o programa com que mais gosto de trabalhar, nesse sentido foi com ele que criei a minha aplicação.

MIT AppInventor, também conhecido como App Inventor for Android, é uma aplicação código aberto originalmente criada pela Google, e atualmente mantida pelo Massachusetts Institute of Technology (MIT).

Ele permite que os recém-chegados à programação de computador criem aplicações de software para o sistema operativo Android. Usa uma interface gráfica que permite aos utilizadores arrastar e soltar objetos visuais para criar uma aplicação que pode ser executada em dispositivos Android. Ao criar o App Inventor, o Google ficou cheio de pesquisas significativas em informática educativa, bem como o trabalho feito dentro do Google em ambientes de desenvolvimento on-line.

A plataforma AppInventor ainda é muito instável e amadora, focada em desenvolver apenas aplicações para pesquisas escolares, impossibilitando o desenvolvimento de qualquer aplicativo profissional.



Imagem 3 - Logotipo do AppInventor

## 3.2. Notepad++

Esta é outra aplicação muito usada em sala de aula, nas aulas de Sistemas de Informação. É uma aplicação gratuita para fazer o download e de fácil aprendizagem em termos de layout da própria aplicação.

Notepad++ é um editor de texto e de código fonte de código aberto sob a licença GPL. Suporta várias linguagens de programação rodando sob o sistema Microsoft Windows (possível utilização no Linux via Wine).

O Notepad++ é distribuído como um software livre. O projeto foi alojado no SourceForge.net, onde foi descarregado mais de 27 milhões de vezes e ganhou duas vezes o premio "SourceForge Community Choice Award" por melhor ferramenta de desenvolvimento. O projeto foi alojado no TuxFamily desde 2010 até 2015 depois foi movido para o GitHub. Ele é baseado no Scintilla, é escrito em C++ utilizando a API Win32 e usa a STL. O objetivo do Notepad++ é oferecer um esguio e eficiente binário com uma interface gráfica totalmente modificável.

As linguagens de programação suportadas pelo Notepad++ são: C, C++, Java, C#, XML, HTML, PHP, JavaScript, makefile, ASCII art, doxygen, ASP, VB/VBScript, Unix Shell Script, BAT, SQL, Objective-C, CSS, Pascal, Perl, Python, Lua, Tcl, Assembly, Ruby, Lisp, Scheme, Smalltalk, PostScript e VHDL. Além disto, os utilizadores podem definir as suas próprias linguagens usando um "sistema de definição de linguagem" integrado, que faz do Notepad++ extensível, para ter realce de sintaxe e compactação de partes de código.

Ele suporta Auto complemento, busca e substituição com integração de expressões regulares, divisão de tela, zoom, favoritos, etc. Tem suporte para macros e plugins. Um plugin de utilizador chamado TextFX, que provê opções de transformação de textos, é incluído por padrão.

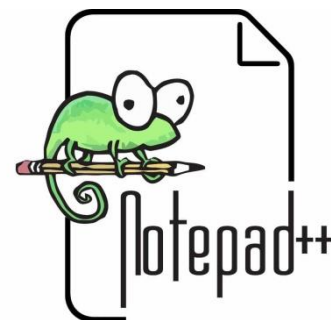


Imagem 4 - Logotipo do Notepad++

### 3.3. SublimText

Aplicação também usada em anos anteriores na disciplina de Sistemas de Informação. Mais propriamente no 11º ano.

Sublime Text é um editor de código-fonte multiplataforma e shareware com uma interface de programação de aplicativos (API) para a linguagem Python. Ele suporta nativamente muitas linguagens de programação e linguagens de marcação, e funções podem ser adicionadas por utilizadores com plug-ins, geralmente criados pela comunidade e mantidos sob licenças de software livre. Inicialmente, o programa foi pensado para ser uma extensão do Vim.



Imagem 5 - Logotipo do SublimText

### 3.4. Microsoft Imagine

Microsoft Imagine, anteriormente conhecido como DreamSpark, é um programa da Microsoft criado para fornecer a estudantes softwares para design e desenvolvimento, sem custos. O programa foi originalmente disponibilizado para os alunos de escolas e universidades na Bielorrússia, Bélgica, Canadá, China, Finlândia, França, Alemanha, Índia, Itália, Marrocos, Países Baixos, Espanha, Suécia, Suíça, Tunísia, Reino Unido e os EUA, mas agora foi expandido para mais de 80 países. Para se inscrever, os estudantes

devem visitar a página do programa e verificar a sua identidade. Se uma instituição não está listada na lista disponível, o utilizador pode verificar manualmente a sua qualificação de estudante, fazendo o upload de um documento comprovativo do vínculo estudantil.

O programa foi anunciado por Bill Gates como DreamSpark em 20 de fevereiro de 2008, durante um discurso na Universidade de Stanford. Estima-se que 35 milhões de alunos serão capazes de obter softwares sem qualquer custo através do programa. O serviço foi renomeado para Microsoft Imagine em 7 de setembro de 2016, para acompanhar a competição estudantil anual, chamada Imagine Cup, realizada pela Microsoft.

A verificação da qualificação de estudante é obrigatória para baixar os programas e obter chaves de produto. No site do Microsoft Imagine, os alunos podem verificar a sua identidade, códigos de acesso fornecidos pelos administradores do programa nas escolas, ou através de e-mails .edu. Os estudantes permanecem registados no programa por cerca de 1 ano. Se os alunos não puderem encontrar as suas respetivas escolas, eles podem, manualmente, enviar um formulário com o comprovativo da sua condição de estudante.



Imagem 6 - Logotipo Microsoft Imagine

### 3.5. Codecademy

Codecademy é um site que possui uma plataforma interativa online e oferece aulas gratuitas de codificação em linguagens de programação como jQuery, Javascript, Python, Ruby, PHP, JAVA bem como as linguagens de marcação, incluindo HTML e CSS. O serviço funciona de maneira que os utilizadores progridam nas lições, avançando com mais códigos progressivamente na medida que se aprende.

Em janeiro de 2014, o site teve mais de 24 milhões de utilizadores que completaram mais de 100 milhões de exercícios. A Codecademy recebeu diversas avaliações positivas de blogs e sites, incluindo o New York Times e a TechCrunch.

Cada indivíduo que entra tem seu próprio perfil. Para motivar os utilizadores a participar, o site oferece feedback e emblemas ao completar exercícios. Há também manuais HTML e CSS disponíveis em cada aula. Além disso, o site permite qualquer pessoa criar e publicar um novo curso usando a ferramenta Criador de Curso.

O Codecademy também disponibiliza um fórum onde entusiastas, programadores principiantes e avançados possam interagir e ajudar-se uns aos outros. Para alguns cursos existem as chamadas 'sandboxes' onde os utilizadores podem testar seu código.

Na Semana da Educação da Ciência do Computador, feita em dezembro de 2013, Codecademy lançou o seu primeiro aplicativo para iOS, chamado de "Hora do Código", destinado a pessoas que querem aprender a programação de uma forma divertida em qualquer lugar. O aplicativo tem foco no básico da programação e inclui o mesmo conteúdo do site.



Imagem 7 - Logotipo do Codecademy



### 3.6. Scratch

Scratch é uma linguagem de programação criada em 2007 pelo Media Lab do MIT. Desde 2013 o Scratch 2 está disponível on-line e como uma aplicação para Windows, OS X, e Linux. O código fonte da versão 1.x está sob a licença GPLv2.

Por não exigir o conhecimento prévio de outras linguagens de programação, ele é ideal para pessoas que estão a começar a programar e foi desenvolvida para ajudar pessoas acima de 8 anos na aprendizagem de conceitos matemáticos e computacionais. Com este programa é possível criar histórias animadas, jogos e outros programas interativos.

Tendo em vista as experiências do Media Lab com a linguagem LOGO, desenvolvida por Papert, o Scratch visa ir além dessa linguagem em três aspetos: torná-la mais suscetiva à manipulação, mais social e mais significativa. Assim a forma como os blocos podem ser manipulados, conferem uma possibilidade de aprendizagem auto-gerida através da prática de manipulação e teste dos projetos. A plataforma online permite que os utilizadores interajam entre si, critiquem e aprendam com os projetos dos outros (remixing). Além disso, o Scratch permite a personalização através da incorporação de imagens e sons externos, bem como a possibilidade de desenhar e gravar som dentro da ferramenta.

O Scratch é considerado mais acessível que linguagens de programação textuais, por se utilizar uma interface gráfica que permite que os programas sejam construídos com blocos encaixados, que faz lembrar o brinquedo Lego. Utiliza uma sintaxe comum a muitas linguagens de programação. É diferente de outras linguagens, não tem nenhum tipo de pontuação obscura.

Cada bloco da linguagem contém um comando em separado, que podem ser agrupados livremente caso se encaixem. E os comandos podem ser modificados através de menus barra de snirks. O Scratch inspirou-se na forma como os DJs fazem a

mistura dos sons para criarem novas músicas. Mas essa linguagem consegue misturar diversos tipos de medias, como imagens, sons e outros programas.

No Scratch existe uma comunidade muito pequena de falantes de língua portuguesa em comparação à quantidade total, mas os poucos que se encontram a programar e a compartilhar esta linguagem de programação, já possuem um conhecimento avançado na ferramenta.

No ano de 2016, o Scratch Foundation anunciou uma parceria com a Google para a nova geração da ferramenta (versão 3.0). Esta nova versão será baseada em tecnologia web nativa (HTML 5) e contará com uma mudança na sua gramática. Os blocos antes montados verticalmente agora serão montados horizontalmente, assim como já ocorre no ScratchJr. O seu lançamento foi no dia 2 de janeiro de 2019.



Imagem 8 - Logotipo do Scratch

### 3.7. Blockly

Blockly é uma biblioteca do lado do cliente para a linguagem de programação JavaScript para a criação de linguagens de programação visual baseadas em bloco (VPLs) e editores. Um projeto do Google, é um software gratuito e de código aberto lançado sob a licença Apache 2.0 . Normalmente é executado num navegador da web e assemelha-se visualmente à linguagem Scratch. Ele também está a ser implementado para os sistemas operativos móveis Android e iOS, embora nem todos os seus recursos baseados em navegador estejam disponíveis nessas plataformas.

O Blockly usa blocos visuais que se vinculam para tornar a escrita de código mais fácil e pode gerar código em JavaScript, Lua, Dart, Python ou PHP. Ele também pode ser personalizado para gerar código em qualquer linguagem de programação textual.



Imagem 9 – Logotipo do Blockly

### 3.8. Cronapp

A Cronapp é uma plataforma de desenvolvimento de software com ambientes na nuvem e on-premise.

O sistema apresenta blocos de programação pré-configurados, que permitem a criação de aplicações e sistemas com os modos de desenvolvimento Low-Code e High-Code, com linguagens Java, Python, Node, entre outras.

Sem tecnologia proprietária, os softwares criados a partir do Cronapp podem ser levados para qualquer ambiente de desenvolvimento. Utiliza drag and drop e, opcionalmente, programação por blocos. Oferece funcionalidades como templates pré-configurados personalizáveis, modelação visual e geração automática de código.

É normalmente indicado para empresas que precisam desenvolver softwares, mobile ou web, com alta produtividade, qualidade e num curto espaço de tempo.



Imagem 10 – Logotipo do Cronapp

### 3.9. Codewars

Codewars, é uma comunidade educacional para programação de computadores. Na plataforma, os programadores de software treinam em desafios de programação conhecidos como kata. Esses exercícios de programação discretos treinam uma variedade de habilidades em uma variedade de linguagens de programação e são concluídos num ambiente de desenvolvimento integrado online. No Codewars, a comunidade e a progressão do desafio são gamificados, com os utilizadores a ganhar classificações e honra por completar kata, contribuir com kata e soluções de qualidade.

A plataforma pertence e é operada pela Qualified, uma empresa de tecnologia que fornece uma plataforma para avaliação e treino de habilidades de engenharia de software.



Imagem 11 – Logotipo do Codewars

### 3.10. Python

Python é uma linguagem de programação de alto nível, interpretada de script, imperativa, orientada a objetos, funcional, de tipagem dinâmica e forte. Foi lançada por Guido van Rossum em 1991. Atualmente, possui um modelo de desenvolvimento comunitário, aberto e gerido pela organização sem fins lucrativos Python Software Foundation. Apesar de várias partes da linguagem possuírem padrões e especificações formais, a linguagem, como um todo, não é formalmente especificada.

A linguagem foi projetada com a filosofia de enfatizar a importância do esforço do programador sobre o esforço computacional. Prioriza a legibilidade do código sobre a velocidade ou expressividade. Combina uma sintaxe concisa e clara com os recursos poderosos de sua biblioteca padrão e por módulos e frameworks desenvolvidos por terceiros.

Python é uma linguagem de propósito geral de alto nível, multiparadigma, suporta o paradigma orientado a objetos, imperativo, funcional e processual. Possui tipagem dinâmica e uma de suas principais características é permitir a fácil leitura do código e exigir poucas linhas de código e compara-se ao mesmo programa em outras linguagens. Devido às suas características, é utilizado, principalmente, para processamento de textos, dados científicos e criação de CGIs para páginas dinâmicas para a web. Foi considerada pelo público a 3ª linguagem "mais amada", de acordo com uma pesquisa conduzida pelo site Stack Overflow em 2018 e está entre as 5 linguagens mais populares, de acordo com uma pesquisa conduzida pela RedMonk.

O nome Python teve a sua origem no grupo humorístico britânico Monty Python, criador do programa Monty Python's Flying Circus, embora muitas pessoas façam associação com o réptil do mesmo nome (em português, píton ou pitão).



Imagem 12 – Logotipo do Python

### 3.11. Mimo

O funcionamento desta ferramenta não poderia ser mais simples: teremos que resolver exercícios de todo tipo para ir avançando nas lições. Estes exercícios são bastante variados: em alguns teremos que escolher a resposta correta dentre várias, por ordem de determinadas respostas, selecionar todas as respostas certa. Mas o verdadeiramente importante é que estão estupendamente explicados e sempre nos dirão as razões de nossos acertos ou erros. São provas que qualquer utilizador, sem importar o nível de conhecimento na matéria, poderá resolver com pouco que leia e assimile os conceitos que se explicam em cada uma das lições.

Mimo é uma aplicação para aprender diferentes linguagens de programação pela comodidade de um smartphone. Talvez não seja tão atrativo no visual como Grasshopper, mas a Mimo conta com muitíssimas linguagens que podemos adicionar à nossa lista de conhecimentos. Uma ferramenta perfeita para qualquer utilizador, sem importar se possuem conhecimentos prévios de programação ou não.



Imagem 13 – Logotipo do Mimo

### 3.12. EasyCoder

Serve essencialmente para aprender programação Java. É uma aplicação inovadora para pessoas que procuram um bom aplicativo de aprendizagem de codificação java, curso de codificação ou tutorial de codificação.

Aprender a programar por meio de uma aplicação tutorial de codificação fácil de entender e começar a criar aplicações ao usar a linguagem de programação Java. O curso de codificação Java é dividido em algumas categorias fundamentais para que se possa aprender sem problemas. É um aplicativo de tutorial em java que começa com a codificação básica e, gradualmente, ensina as coisas de nível mais avançado. Se está procurando um aplicativo para programar java, que tenha o curso completo de aprendizagem de codificação, o Easy Coder é perfeito.

O aplicativo oferece tutoriais em vídeo e questionários relacionados ao Java. Ele ensina todos os fundamentos relacionados a uma tecnologia específica para que se possa começar a programar. O aplicativo ensina codificação/programação de uma forma divertida para que não fique entediado. Aplicativos de aprendizagem de código nunca foram tão incríveis. E dizem que este é o melhor aplicativo de aprendizagem de Java para Android online.



Imagem 14 – Logotipo do Easy Coder



## Capítulo IV – Desenvolvimento do Projeto

## 4.1. Aplicação utilizada

A aplicação que utilizei para criar o meu jogo foi o “APPINVENTOR”, aplicação essa que foi matéria lecionada nas aulas de PPM, então aproveitei os conhecimentos que adquiri nessas aulas e programei o meu jogo.

A aplicação é online e de fácil aprendizagem, tem ferramentas muito básicas e bem nítidas.

## 4.2. Barras de Ferramentas e Algumas Funcionalidades

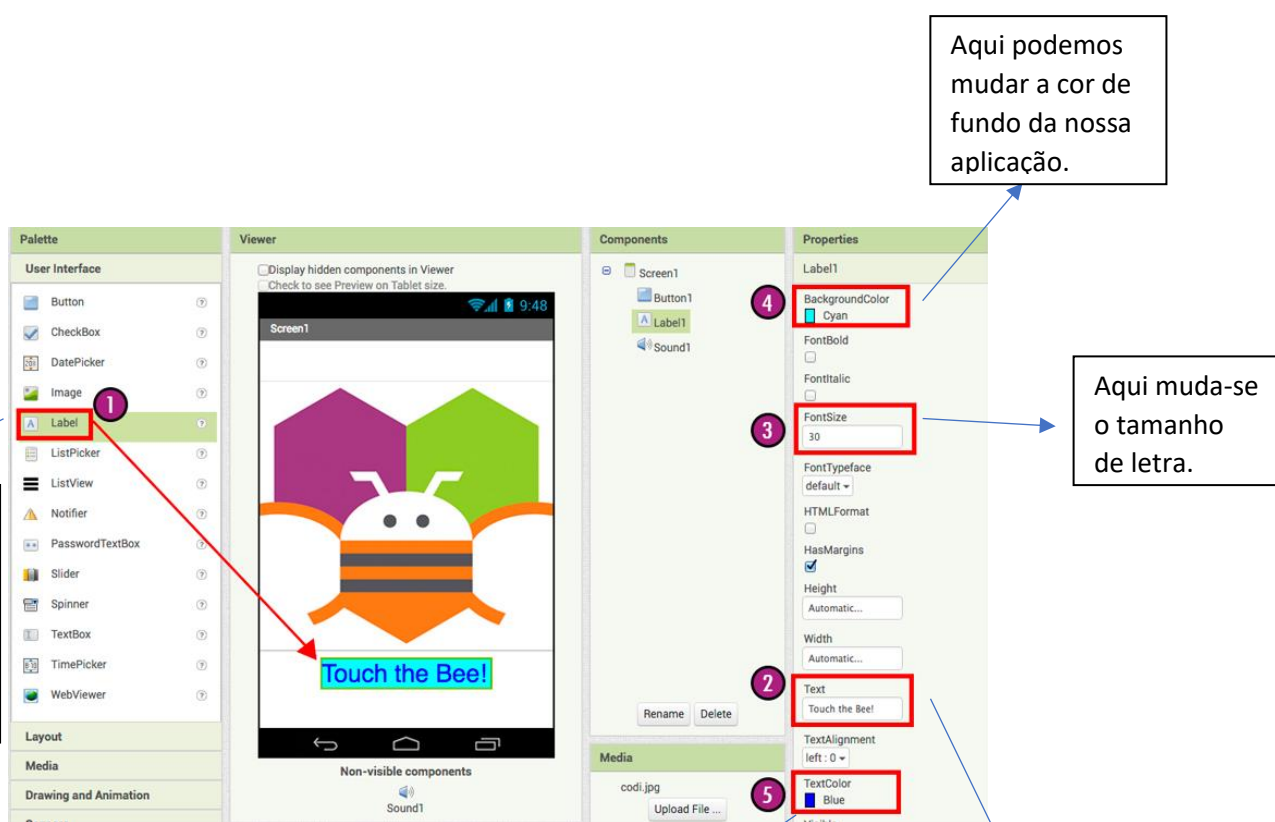


Imagem 15 - Explicação das funcionalidades do AppInventor

### 4.3. Código da programação

```
quando Temporizador1 .Disparo
fazer
  chamar controla_bola
  chamar controla_telemovel

quando bola .AlcançouABorda
  borda
  fazer
    se obter borda = 1
    então se obter global dir_y_bola < 0
    então ajustar global dir_y_bola para obter global dir_y_bola x -1
    senão, se obter borda = -1
    então se obter global dir_y_bola > 0
    então ajustar global dir_y_bola para obter global dir_y_bola x -1

quando bola .ColidiuCom
  outro
  fazer
    se obter outro = barra_jogador ou obter outro = barra_telemovel
    então
      ajustar global dir_x_bola para obter global dir_x_bola x -1
      ajustar global dir_y_bola para
        obter global dir_y_bola + (bola .Y + bola .Altura / 2 - SpriteImagem .Y do componente obter outro + SpriteImagem .Altura do componente obter outro)

para controla_telemovel
fazer
  se bola .X > Pintura1 .Largura / 2
  então se
    obter global dir_x_bola > 0
    obter global dir_x_bola + (bola .Y + bola .Altura / 2 - barra_telemovel .Y + barra_telemovel .Altura / 2) > 0
    obter global dir_x_bola < 0
    obter global dir_x_bola + (bola .Y + bola .Altura / 2 - barra_telemovel .Y + barra_telemovel .Altura / 2) < 0

quando Screen2 .Inicializar
fazer
  ajustar Legenda1 .Texto para select list item list obter global questoes index 1
  ajustar VisualizadorDeListas1 .Elementos para select list item list obter global itens index 1
  ajustar Imagem1 .Imagem para select list item list obter global imagens_das_questoes index 1

quando ok .Clique
fazer
  abrir outro ecrã nomeDoEcrã " Screen1 "
```

Imagem 16 - Código do jogo

#### 4.4. Jogo Final

Depois de muito tempo investido e muito esforço, finalmente finalizei o jogo.

*Ping Pong* é um jogo eletrónico de desporto em duas dimensões que simula ténis de mesa. O jogador controla uma paleta (barra vertical) no jogo movendo-a verticalmente no lado esquerdo da tela, e compete contra o telemóvel que controla uma segunda barra vertical no lado oposto.

O objetivo é o jogador usar a paleta para acertar na esfera (bola) e mandá-la para o outro lado.

A paleta é dividida em oito segmentos, com o segmento central retornando a bola num ângulo de 90º em relação à paleta e os segmentos externos retornando a bola em ângulos cada vez menores. A bola aumenta de velocidade cada vez que é rebatida, reiniciando a velocidade caso algum dos jogadores não acerte a bola. O objetivo é fazer mais pontos que seu oponente, fazendo com que o oponente não consiga retornar a bola para o outro lado.

Antes de iniciar o jogo o jogador deve clicar no botão amarelo para ler as instruções.

Depois deve clicar no botão Começar e o jogo inicia.

Do lado esquerdo o jogador visualiza os pontos obtidos, e no lado direito visualiza os pontos do telemóvel.

O jogador pode a qualquer momento clicar no botão Pergunta e ser-lhe-á feita uma pergunta sobre Pinhel, à qual deve responder.

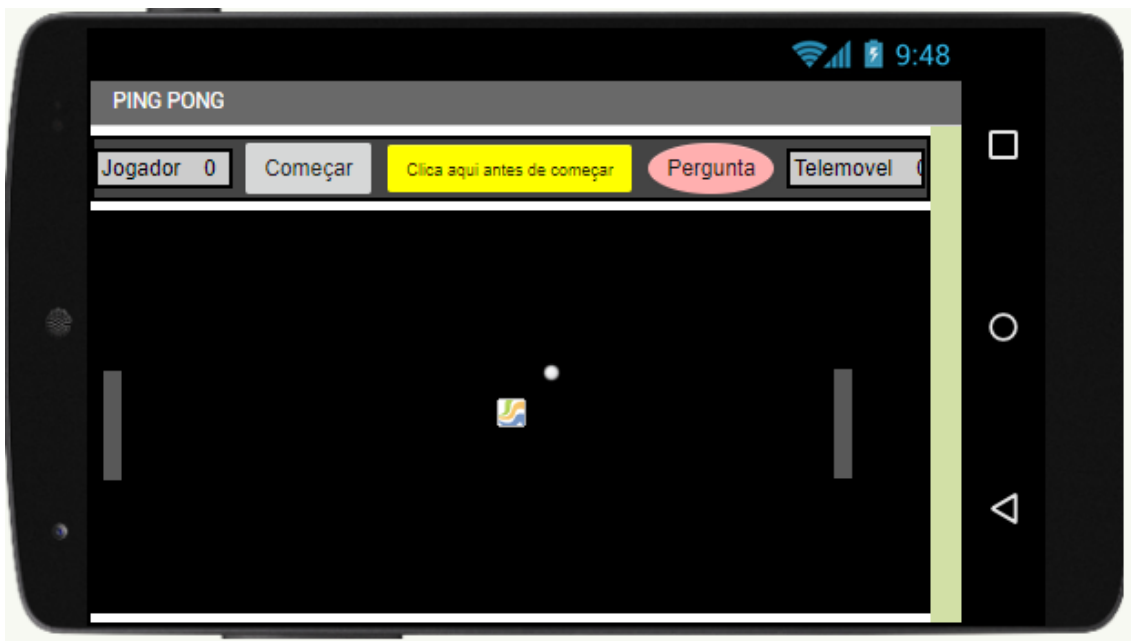


Imagem 17 - Jogo Finalizado

## 4.5. Divulgação do jogo

A ideia inicial era divulgar o jogo na Feira das Tradições, no entanto, devido à pandemia não foi possível participar nem expor a aplicação na Feira das Tradições. São coisas que ninguém consegue controlar, e por causa desta pandemia as minhas ideias acabaram um pouco por não terem sido postas em prática.

No entanto, como a evolução da pandemia evoluiu positivamente, no dia 17 de junho, realizou-se na nossa escola o Dia da Informática. Aproveitei então para divulgar e apresentar o meu jogo aos alunos participantes. Estes acharam o meu jogo muito divertido e interessante, tendo ficado motivados para seguirem os estudos na área profissional, mais propriamente no Curso Profissional de Multimédia.



Imagem 18 – Alunos no Dia da Informática a jogarem no Tablet

## Capítulo V – Conclusões



## 5.1. Análise Crítica

Começo por referir que todos os prazos e objetivos foram alcançados o que me faz gerar um sentimento de satisfação no que se refere ao trabalho realizado.

Ao longo dos três anos de curso aprendi muita coisa, óbvio que tive dificuldades, mas que superei com a ajuda dos meus professores e colegas, tudo isto conectado à minha motivação e ao facto de nunca desistir quando um obstáculo se aproxima.

Destaco também as muitas horas de trabalho investido ao longo do ano através de um trabalho contínuo e progressivo.

## 5.2. Autoavaliação

Pelo que realizei, horas gastas, esforço e muita dedicação, faço um balanço muito positivo do trabalho realizado.

### 5.3. Conclusão

A aquisição de novas competências foi uma constante ao longo destes 3 anos, que culminou na elaboração desta PAP. Além de colocar em prática muitos conhecimentos abordados ao longo do curso, trouxe-me a possibilidade de adquirir novas competências na área da programação.

Não posso também deixar de expressar o meu agradecimento a todos os intervenientes que diretamente ou indiretamente estiveram envolvidos neste projeto, destacando o forte apoio dos meus colegas de turma e de todos os professores que partilharam os seus conhecimentos ao longo destes três anos.

Para concluir gostei de realizar muito esta PAP principalmente porque foi uma ótima experiência de aprendizagem, podendo ser uma mais-valia para um futuro no mercado de trabalho.

## Webgrafia

História do videojogo do Ping Pong; <https://pt.wikipedia.org/wiki/Pong>; 03/11/2020.

Jogo Pong; <https://pt.wikipedia.org/wiki/Pong>; 02/03/2021.

Conceitos de Programação; <https://idocode.com.br/blog/programacao/o-que-e-programacao/>; 9/06/2021.

Cuidados ao criar um jogo de telemóvel; <https://www.devmedia.com.br/criando-jogos-para-celular-a-partir-do-estudo-de-jogos-antigos/8327>; 11/06/2021.