

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

HUGO BEVILAQUA DE CARVALHO REIS

JOGOS DE TIMELINE: ELABORAÇÃO E DESENVOLVIMENTO DE UM JOGO
EDUCATIVO PARA ENSINO DO PROCESSO DE DESCOBERTA E
DESENVOLVIMENTO DE FÁRMACOS

RIO DE JANEIRO
2023

HUGO BEVILAQUA DE CARVALHO REIS

JOGOS DE TIMELINE: ELABORAÇÃO E DESENVOLVIMENTO DE UM JOGO
EDUCATIVO PARA ENSINO DO PROCESSO DE DESCOBERTA E
DESENVOLVIMENTO DE FÁRMACOS

Trabalho de conclusão de curso de graduação
apresentado ao Instituto de Computação da
Universidade Federal do Rio de Janeiro como
parte dos requisitos para obtenção do grau de
Bacharel em Ciência da Computação.

Orientador: Prof. Geraldo Bonorino Xexéo

RIO DE JANEIRO

2023

CIP - Catalogação na Publicação

R375j Reis, Hugo Bevilaqua de Carvalho
 Jogos de timeline: elaboração e desenvolvimento
de um jogo educativo para ensino do processo de
Descoberta e Desenvolvimento de fármacos / Hugo
Bevilaqua de Carvalho Reis. -- Rio de Janeiro, 2023.
 73 f.

 Orientador: Geraldo Bonorino Xexéo.
Trabalho de conclusão de curso (graduação) -
Universidade Federal do Rio de Janeiro, Instituto
de Computação, Bacharel em Ciência da Computação,
2023.

 1. Jogos educativos. 2. Desenvolvimento de
jogos. 3. Linhas do tempo. I. Xexéo, Geraldo
Bonorino, orient. II. Título.

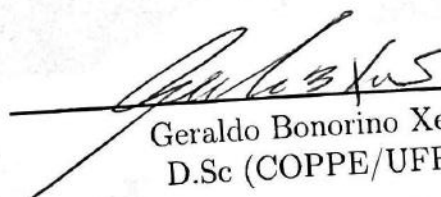
HUGO BEVILAQUA DE CARVALHO REIS

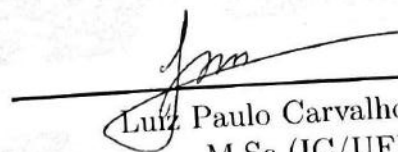
JOGOS DE TIMELINE: ELABORAÇÃO E DESENVOLVIMENTO DE UM JOGO
EDUCATIVO PARA ENSINO DO PROCESSO DE DESCOBERTA E
DESENVOLVIMENTO DE FÁRMACOS

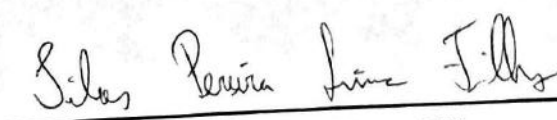
Trabalho de conclusão de curso de graduação
apresentado ao Instituto de Computação da
Universidade Federal do Rio de Janeiro como
parte dos requisitos para obtenção do grau de
Bacharel em Ciência da Computação.

Aprovado em 11 de setembro de 2023

BANCA EXAMINADORA:


Geraldo Bonorino Xexéo
D.Sc (COPPE/UFRJ)


Luiz Paulo Carvalho da Silva
M.Sc (IC/UFRJ)


Silas Pereira Lima Filho
D.Sc (IC/UFRJ)

Dedico este trabalho à minha família, sem o suporte da qual a realização dele não teria sido possível.

AGRADECIMENTOS

Agradeço ao professor Xexéo, meu orientador para a realização deste trabalho. Agradeço também ao corpo docente da UFRJ, que me proporcionou o aprendizado necessário para realizá-lo.

"The two most powerful warriors are patience and time."

Leo Tolstoy

RESUMO

Este trabalho aborda o design e desenvolvimento de jogos no estilo *timeline*, bem como a implementação de um jogo educacional deste mesmo tipo que tem como tema e objetivo o ensino do processo de Descoberta e Desenvolvimento de Fármacos e medicamentos (DDF). Um jogo *timeline* é um jogo de cartas onde cada carta representa um evento atribuído de uma data, e cabe ao jogador ordená-las cronologicamente da maneira que acredita ser correta. O jogo implementado chama-se Screener Timeline, uma versão digital e de jogador único do jogo de tabuleiro multijogador SCREENER, desenvolvida por meio do motor de jogo Unity e scripts em C#.

Palavras-chave: timeline; jogo educacional; design; desenvolvimento.

ABSTRACT

This work addresses the design and development of timeline-styled games, as well as the implementation of an educational game of the same type, which has as its theme and objective the teaching of the process of Discovery and Development of Pharmaceuticals and Medicine. A timeline game is a card game wherein each card represents an event assigned a date, and the player is tasked with chronologically ordering said cards in the manner they believe to be correct. The implemented game is named Screener Timeline, a digital, single-player version of the multiplayer tabletop game SCREENER, developed by means of the Unity game engine and C# scripting.

Keywords: timeline; educational game; design; development.

LISTA DE ILUSTRAÇÕES

Figura 1 – Tabuleiro de Senet, Egito, c. 1390 - 1353 AEC	15
Figura 2 – SCREENER, Anúncio	16
Figura 3 – G1: Quando Aconteceu? Especial Copa do Mundo	16
Figura 4 – Carta de tarefa do SCREENER, Etapa 1, Tarefa b	21
Figura 5 – Scriptable Object nomeado <i>Step 1</i> , representando a primeira tarefa do jogo	32
Figura 6 – Configuração de um componente grid	32
Figura 7 – Zona <i>mesa</i> suprida de 4 objetos-filho alinhados a sua grid, e zona <i>mão</i> suprida de 1 objeto alinhado a sua grid	33
Figura 8 – Configuração do componente <i>event trigger</i> para movimentar um objeto	34
Figura 9 – <i>Prefab</i> elaborada para uma carta genérica	35
Figura 10 – Matriz de colisão do projeto	37
Figura 11 – Variáveis públicas do script de jogo	39
Figura 12 – Partida em progresso	43
Figura 13 – Janela de fim de jogo	45
Figura 14 – Partida em progresso, com utilização de slider	48
Figura 15 – Carta de poder do SCREENER	50
Figura 16 – Plano de fundo do jogo criado, adaptado do jogo original	50
Figura 17 – Ícone do jogo em ambiente android	51
Figura 18 – Menu principal do Screener Timeline	51
Figura 19 – Tela de informações do Screener Timeline	52
Figura 20 – Tela de jogo do Screener Timeline	52
Figura 21 – Término de uma partida do Screener Timeline	53
Figura 22 – Página de download do Screener Timeline	54
Figura 23 – Resposta do formulário de avaliação: Data de nascimento	62
Figura 24 – Resposta do formulário de avaliação: Cidade de residência	62
Figura 25 – Resposta do formulário de avaliação: Estado de residência	63
Figura 26 – Resposta do formulário de avaliação: Gênero	63
Figura 27 – Resposta do formulário de avaliação: Idade	63
Figura 28 – Resposta do formulário de avaliação: Raça	64
Figura 29 – Resposta do formulário de avaliação: Formação	64
Figura 30 – Resposta do formulário de avaliação: Titulação acadêmica	64
Figura 31 – Resposta do formulário de avaliação: Renda	65
Figura 32 – Resposta do formulário de avaliação: Conhecimento prévio de jogos educacionais	65

Figura 33 – Resposta do formulário de avaliação: Interação prévia com jogos educacionais	65
Figura 34 – Resposta do formulário de avaliação: Diversão com jogos educacionais .	66
Figura 35 – Resposta do formulário de avaliação: Plataformas de jogo	66
Figura 36 – Resposta do formulário de avaliação: Frequência de jogo	66
Figura 37 – Resposta do formulário de avaliação: Avaliação da Reação	67

LISTA DE CÓDIGOS

Código 1	Criação do Scriptable Object para cartas	31
Código 2	Script para movimentação drag-and-drop de objetos	33
Código 3	Script para display das informações das cartas	36
Código 4	Script de movimentação com adição de sistema para detectar colisões e alinhar cartas à zona <i>mesa</i>	38
Código 5	Script de jogo com instanciação adequada das cartas	40
Código 6	Função <i>Start()</i> do script de jogo modificado para avaliação de cartas	41
Código 7	Função de avaliação de cartas posicionadas	42
Código 8	Função <i>Update()</i> do script de jogo modificado para avaliação de cartas	42
Código 9	Função <i>Update()</i> do script de jogo modificado	44
Código 10	Função de fim de jogo <i>GameOver()</i>	46
Código 11	Script contador de tempo	47
Código 12	Script para movimentação da zona <i>mesa</i> via slider	48
Código 13	Script para transição de telas	49

LISTA DE QUADROS

Quadro 1 – SGDD: Screener Timeline	61
--	----

LISTA DE ABREVIATURAS E SIGLAS

DDF	Descoberta e Desenvolvimento de Fármacos e medicamento
Fig.	Figura
c.	circa
AEC	Antes da Era Comum
Obj.	Objeto
SGGD	Short game design document
FDA	Food and Drug Administration
NDA	New Drug Application

SUMÁRIO

1	INTRODUÇÃO	15
1.1	MOTIVAÇÃO	15
1.2	OBJETIVOS	16
1.3	PROCESSO DE DESENVOLVIMENTO	17
2	SCREENER E TIMELINE	18
2.1	JOGOS EDUCACIONAIS	18
2.2	FUNCIONAMENTO: POR QUÊ TIMELINE?	19
2.3	SCREENER E SEU FUNCIONAMENTO	20
2.4	SINERGIA E ALTERAÇÕES	22
2.5	O PROCESSO DE DDF	23
2.6	TRABALHOS CORRELATOS	25
3	ELABORAÇÃO	27
3.1	PLANEJAMENTO DO JOGO	27
3.2	REQUERIMENTOS PARA O FUNCIONAMENTO DO JOGO	28
3.3	FERRAMENTAS ESCOLHIDAS	29
4	DESENVOLVIMENTO	31
4.1	INFORMAÇÃO DAS CARTAS	31
4.2	DEFINIÇÃO DE ÁREAS PARA CARTAS	31
4.3	MOVIMENTAÇÃO DAS CARTAS	32
4.4	DISPLAY DAS CARTAS	34
4.5	DETECÇÃO DE COLISÃO ENTRE CARTAS E ÁREAS	36
4.6	INSTANCIAMENTO DAS CARTAS	39
4.7	AVALIAÇÃO DE CARTAS POSICIONADAS E SUBSEQUENTE MUDANÇA DE COR	41
4.8	DETECÇÃO DE FIM DE JOGO E CONTADOR DE ERROS	43
4.9	FIM DE JOGO	45
4.10	CONTADOR DE TEMPO	46
4.11	SLIDER PARA CARTAS SOBRE A MESA	47
4.12	TRANSIÇÃO ENTRE TELAS	49
4.13	DESIGN VISUAL	49
4.14	RESULTADO: SCREENER TIMELINE	50
5	AVALIAÇÃO	54

5.1	QUESTIONÁRIO	54
5.2	MELHORIAS	54
6	CONCLUSÃO	57
	REFERÊNCIAS	58
	GLOSSÁRIO	60
	APÊNDICE A – SHORT GAME DESIGN DOCUMENT: SCREENER TIMELINE	61
	APÊNDICE B – RESPOSTAS DO FORMULÁRIO DE AVALIAÇÃO DO SCREENER TIMELINE	62
	ANEXO A – PERGUNTAS DO FORMULÁRIO DE AVALIAÇÃO DO SCREENER TIMELINE	68

1 INTRODUÇÃO

Ao longo da história humana, jogos serviram diversos papéis, incluindo mas não limitado a confraternização, entretenimento, e desenvolvimento de habilidades e competências. Com o passar do tempo, jogos e suas características sofreram mudanças, adaptações e cresceram em proeminência na sociedade humana, de jogos de tabuleiro datados de milênios no passado (Senet, Figura 1), até os jogos digitais compactos contemporâneos.

Figura 1 – Tabuleiro de Senet, Egito, c. 1390 - 1353 AEC



Fonte: Brooklyn Museum, Charles Edwin Wilbour Fund (Obj. 49.56a-b)

Em consequência destas mudanças, em tempos recentes o desenvolvimento de jogos educacionais surgiu como uma tática inovadora para atrair e manter a atenção de estudantes e promover a experiência de aprendizado de maneira lúdica e envolvente. Jogos educacionais transformam o processo de aprendizado, fazendo dele uma experiência interativa e dinâmica. Ao estimular a motivação natural do jogador quanto a desafios e solução de problemas, essa modalidade de jogo permite ao estudante aprender novos temas, praticar seu conhecimento, e desenvolver habilidades lógicas importantes, estando ainda ativamente envolvido no processo de aprendizagem.

1.1 MOTIVAÇÃO

Tratando-se de jogos educacionais, um exemplo é o SCREENER (Figura 2), um jogo que tem como finalidade o ensino do processo de Descoberta e Desenvolvimento de Fármacos e medicamentos (DDF). Produto dos esforços de uma equipe multidisciplinar, o SCREENER foi desenvolvido a partir de uma proposta do Prof. François Noël em colaboração com o laboratório LUDES (NOEL et al., 2021).

Figura 2 – SCREENER, Anúncio



Fonte: <https://www.screener.com.br>

O jogo obteve sucesso no seu objetivo de proporcionar um aprendizado lúdico sobre o assunto, e já foi usado por diversos cursos de Pós-Graduação em Farmacologia e outras áreas. Consequentemente, ele serve também como a inspiração para este trabalho. Sendo ele um jogo de tabuleiro para até seis jogadores, a motivação para este projeto é criar uma versão que abranja um público ainda não alcançado pelo jogo original, assim continuando a disseminar o conhecimento que ele busca promover.

1.2 OBJETIVOS

Com relação a assuntos como o processo de DDF, que envolvem uma sequência de eventos, jogos do tipo timeline (Figura 3) tem relevância particular. Esses são jogos que encarregam o jogador de organizar cronologicamente uma série de eventos, assim buscando ensinar um dado assunto e promover a capacidade de pensamento crítico.

Figura 3 – G1: Quando Aconteceu? Especial Copa do Mundo



Fonte: <https://especiais.g1.globo.com/mundo/copa-do-catar/2022/quando-aconteceu>

Mantendo isso em mente, este trabalho tem como objetivo estabelecer métodos e requi-

sitos para o design e desenvolvimento de jogos timeline, e subsequentemente implementar um jogo digital de jogador único desse mesmo tipo, que faça uso das particularidades da modalidade para propiciar o mesmo aprendizado lúdico do SCREENER original de maneira compacta e interativa.

1.3 PROCESSO DE DESENVOLVIMENTO

O processo de desenvolvimento deste projeto consiste no seguinte trajeto. Primeiro, analisa-se o funcionamento do SCREENER original e determina-se que aspectos dele devem ser mantidos e quais devem sofrer alterações para melhor adequação ao formato novo. A seguir, define-se que funcionalidades são necessárias para a execução de um jogo timeline, e quais ferramentas usar para implementá-las. Por fim, ocorre o desenvolvimento do jogo. Estas etapas serão subsequentemente abordadas individualmente.

2 SCREENER E TIMELINE

Com o intuito de viabilizar o desenvolvimento adequado do jogo, esta seção examina os pontos fortes da modalidade timeline de jogo, o funcionamento do SCREENER, as características dele que são aptas a ser adaptadas para o formato, e as características que podem ou devem sofrer alterações. Esse planejamento busca garantir a qualidade da concepção inicial do jogo, assim mantendo os aspectos do SCREENER original que apresentam sinergia com o novo formato, e simultaneamente atingindo o mesmo objetivo de aprendizado interativo.

2.1 JOGOS EDUCACIONAIS

Jogos educacionais são atividades lúdicas desenvolvidas primariamente para propósitos educacionais. Eles são um subtipo de *jogos sérios*, jogos que contribuem para um objetivo além de apenas entretenimento. Jogos educacionais são criados com o objetivo de disseminar informação, ensinar conhecimentos específicos, ou promover aprendizagem de maneira divertida, assim capturando a atenção do jogador (BACKLUND; HENDRIX, 2013). Ao combinar o conteúdo educacional com uma atividade lúdica, jogos educacionais fazem a experiência de aprendizado mais efetiva e agradável. A *gamificação*, a aplicação de elementos de jogos em outros ambientes¹, tem resultados comprovados em ambientes educacionais, particularmente em ambientes relacionados à educação farmacêutica, como o SCREENER (LOUNSBURY et al., 2022). Eles são efetivos em auxiliar a atenção, confiança, comunicação, motivação e aprendizado dos estudantes.

Ademais, jogos educacionais têm um impacto significante na habilidade de solucionar problemas do jogador. O processo de aprendizagem compreende tanto a obtenção de conhecimento como também a habilidade de saber quando e como utilizá-lo. Conseqüentemente, a solução de problemas compreende o desenvolvimento das seguintes habilidades (KALMPOURTZIS, 2018):

- Raciocínio;
- Comunicação;
- Observação;
- Análise;
- Avaliação;

¹ Alguns autores usam o termo gamificação de forma ampla, indicando todo o uso de jogos em educação, outros usam de forma restrita, indicando apenas o uso de elementos dos jogos em atividades que não são jogos

- Identificação de conhecimento existente.

Em particular para assuntos de conteúdo extenso e variado, o aprendizado baseado em jogos (*game-based learning*) é de interesse. Aplicar métodos de ensino adequados e efetivos para assuntos complexos pode se provar desafiador às instituições de ensino, e em consequência disso, jogos educacionais são uma solução elegante para esse dilema. Eles são uma ferramenta flexível, podendo fazer uso de um nível tecnológico baixo, como jogos de tabuleiro, ou níveis mais avançados, como jogos eletrônicos e simulações digitais. Aprendizado baseado em jogos pode se adequar a qualquer tópico, e promove o engajamento e interesse dos estudantes, especialmente no caso de conteúdo considerado repetitivo ou de pouco interesse (OESTREICH; GUY, 2022).

Jogos educacionais tem o poder de mudar a representação e formato de problemas a serem resolvidos por seus jogadores, assim proporcionando a eles a oportunidade de elaborar suas próprias estratégias para solucionar e lidar com desafios. Da mesma maneira que uma pessoa jogando dominó não considera trabalhoso examinar as peças e contar e combinar os valores de cada uma, pode-se representar qualquer tema de maneira lúdica em um jogo, assim motivando o jogador a adquirir conhecimento de maneira natural, intuitiva e interativa (KALMPOURTZIS, 2018).

2.2 FUNCIONAMENTO: POR QUÊ TIMELINE?

Um jogo timeline típico consiste num baralho de cartas que representam eventos sequenciais, e sua utilização segue a seguinte estrutura:

- Para iniciar o jogo, uma carta aleatória é colocada sobre a mesa, com informações visíveis sobre sua posição em uma cronologia, como uma data, ou outro método de ordenação;
- O jogador recebe uma carta aleatória, sem a informação de ordenação;
- O jogador coloca a carta sobre a mesa, posicionada antes, após, ou entre as cartas já jogadas, de acordo com sua suposição de ordenação;
- Avalia-se e revela-se se a carta foi posicionada de maneira correta com relação às outras cartas sobre a mesa;
- O processo é repetido até que todas as cartas sejam posicionadas ou, alternativamente, até que uma dada quantidade de erros seja cometida.

Quando as cartas são físicas, normalmente um lado tem a data e o outro não. Quando são digitais, a informação da data é simplesmente omitida até que a carta digital seja posicionada.

Não obstante possíveis diferenças ou adições presentes em jogos individuais, dessa estrutura geral pode-se inferir que esta modalidade de jogo apresenta as seguintes características:

- **Promove entendimento cronológico:** Ao organizar os eventos em ordem sequencial, jogadores obtêm conhecimento aprofundado quanto a quando e como cada evento ocorre (BUTLER, 2017);
- **Encoraja pensamento crítico:** A necessidade de avaliar o contexto e significância dos eventos representados em cada carta treina as habilidades de análise e inferência do jogador (KARBALAEI, 2012);
- **Apresenta uma representação visual coerente:** A apresentação clara provida pelo jogo auxilia a visualização do fluxo de eventos e elucida as relações entre eles (SILVA; CATARCI, 2000);
- **Treina memória e retenção de informação:** Organizar eventos em ordem sequencial e tentar recordar a sequência original implica prática de recuperação, ou *retrieval practice* (BERTILSSON et al., 2020), uma estratégia de estudo que promove aprendizagem eficaz por meio de auto-avaliação;
- **Proporciona formação de conexões:** A representação em timeline permite ao jogador mais facilmente identificar conexões e contexto entre eventos, o que proporciona uma compreensão mais abrangente deles (ZHUSSUPOVA; KALIZHANOVA, 2021);
- **Facilita integração educacional:** Jogos timeline são compactos, tem curta duração, e tipicamente não necessitam de múltiplas pessoas para serem jogados. Isso permite que eles complementem métodos de ensino tradicionais e ofereçam integração simples com ambientes educacionais como salas de aula.

Por esses motivos, o formato de timeline é de interesse para a elaboração do jogo.

2.3 SCREENER E SEU FUNCIONAMENTO

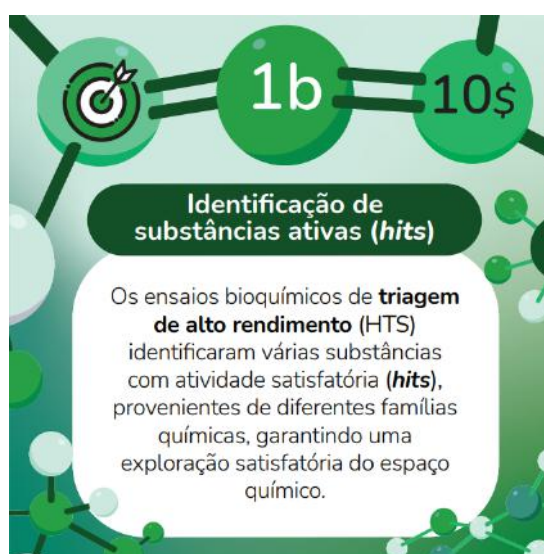
O SCREENER apresenta uma abstração do processo de Descoberta e Desenvolvimento de Fármacos no formato de um jogo de tabuleiro. Os jogadores cooperam para coletar, em ordem sequencial, cartas representando as tarefas que constituem cada etapa do processo, mas simultaneamente também competem para coletar individualmente a maior quantidade de cartas.

O jogo ocorre mediante os seguintes componentes:

- **1 tabuleiro:** Acomoda as cartas e define o espaço do jogo;

- **1 mapa do processo:** Indica as 7 etapas que compõem as fases do processo de DDF;
- **Um conjunto de cartas;**
 - 1 carta de início: Posição inicial dos jogadores no tabuleiro;
 - 2 cartas FDA: Marcam momentos de pedido à FDA (Food and Drug Administration);
 - 29 cartas de tarefa (figura 4): Representam as tarefas contidas em cada etapa do processo de DDF ;
 - 58 cartas de bônus/revés: Eventos durante o jogo que podem ou não beneficiar o jogador;
 - 6 cartas de poder: Uma por jogador, podem ser usadas 1 vez por jogo como auxílio;
- **Notas de dinheiro:** Usadas para comprar cartas de tarefa, e ocasionalmente para desempate;
- **6 peões:** Representam os jogadores;
- **1 dado de 6 faces:** Movimentação dos peões;
- **31 chatons de acrílico:** Indicam o número total de cartas adquiridas durante o jogo, determinam o vencedor.

Figura 4 – Carta de tarefa do SCREENER, Etapa 1, Tarefa b



Fonte: Materiais para Download do SCREENER, Cartas - Coloridas (screener.com.br/#download)

Com estes elementos em consideração, uma partida do jogo é conduzida da seguinte maneira:

- **Preparação**

- A carta de início é colocada no centro do tabuleiro, e as cartas FDA são colocadas sob ela, no mesmo espaço;
- As cartas de tarefa são embaralhadas e distribuídas sobre o tabuleiro com o verso para cima, formando o espaço por onde os jogadores se movimentarão;
- O mapa do processo é colocado sobre a mesa, próximo ao tabuleiro;
- As cartas de bônus/revés são separadas em 7 pilhas sobre a mesa, e cada pilha é embaralhada individualmente;
- Cada jogador recebe \$27 em notas de dinheiro;
- Cada jogador escolhe um peão, e coloca-o sobre a carta de início;

- **Jogo**

- O primeiro jogador lança o dado;
 - * Se 1 - 5, anda com seu peão o valor obtido de casas ou menos;
 - * Se 6, pega uma carta de bônus/revés, age de acordo com a descrição dela, e passa sua vez;
- Se andou e parou sobre o verso de uma carta da etapa atual, ela é virada;
- Se virou uma carta da etapa atual, ou se parou sobre uma já virada, o jogador tem a opção de usar seu dinheiro para comprá-la, se ela for a tarefa da vez;
- Se ela não for a tarefa da vez, o jogador recebe o valor da carta em dinheiro;
- Ao comprar uma carta, o jogador discute o conteúdo dela com os outros jogadores, e recebe um chaton;
- O turno do jogador acaba, e o próximo jogador repete o processo;

Por fim, o vencedor é aquele que obteve mais cartas no total, assim obtendo a maior quantidade de chatons. O aprendizado proporcionado pelo jogo acontece predominantemente durante a etapa imediatamente após um jogador comprar uma carta, quando o conteúdo dela é discutido entre os jogadores.

2.4 SINERGIA E ALTERAÇÕES

Com as observações realizadas nas seções 2.2 e 2.3, é evidente que o SCREENER difere consideravelmente de um jogo timeline típico. Ele é um jogo multijogador, faz uso de um tabuleiro e diversas peças auxiliares, apresenta um sistema monetário, entre

outras características. Isso não obstante, o jogo revolve em torno das etapas e tarefas componentes de DDF, um processo sequencial, e portanto há elementos dele que podem ser de uso a um jogo na modalidade timeline. É necessário, então, identificar que alterações são necessárias para a adaptação do SCREENER.

Em particular, as seguintes medidas são deliberadas para a elaboração do dito jogo:

- **Escolha do tema:** Um jogo timeline necessita um assunto adequado à representação cronológica. Este é, é claro, o processo de Descoberta e Desenvolvimento de Fármacos;
- **Identificação de eventos principais:** Uma vez determinado o tema, é necessário identificar os aspectos de maior importância presentes nele. O processo de DDF pode ser dividido em 7 etapas principais. Essas etapas são escolhidas para compreender o conteúdo educacional que irá constituir o baralho do jogo;
- **Definição das cartas de evento:** O jogo é constituído por um baralho de cartas representando eventos sequenciais, e portanto é necessário determinar quantas e qual será o conteúdo preciso dessas cartas. Para servir esse propósito, as 29 cartas de tarefa do SCREENER serão adaptadas, cada uma representando uma tarefa componente de uma das 7 etapas do processo de DDF;
- **Modificação dos componentes de jogo:** O SCREENER faz uso de diversas peças auxiliares, incluindo mas não limitado a um tabuleiro, chatons e peões. Estas são ou desnecessárias para o funcionamento de um jogo timeline, ou redundantes no caso de um jogo digital, e portanto não serão utilizadas;
- **Ideação de design visual:** É ideal uma apresentação visual que remeta o jogador ao SCREENER original, mas que concomitantemente não detraia do formato novo. Neste intuito, faz-se uso dos padrões de cor e aparência do jogo original para elaborar a representação visual da adaptação de maneira adequada ao intento original.

2.5 O PROCESSO DE DDF

O processo de Descoberta e Desenvolvimento de Fármacos é extenso e de custo extremamente caro (NOEL; XEXEO, 2021). Ademais, apesar de custar em média mais de um bilhão de dólares e frequentemente apresentar duração de mais de uma década, ele raramente resulta em sucesso. Não há consenso na comunidade científica com relação à terminologia ou quanto à melhor maneira de representar a sequência de etapas multidisciplinares que constituem esse procedimento. Ressalta-se que a interseção entre Descoberta e Desenvolvimento ocorre na eleição do candidato à fármaco, sendo requerida a autorização da agência reguladora para o início dos estudos clínicos. Adicionalmente, em prática

o processo apresenta variações dependendo dos objetivos e estratégias empregadas por cada empresa, e não é incomum que tarefas ocorram de maneira simultânea.

De interesse particular ao propósito deste trabalho é a estrutura de 7 etapas subdivididas em 29 tarefas sequenciais que o SCREENER usa para representar o processo:

- **Etapa 1: Identificação de substâncias ativas**
 - Tarefa 1a: Identificação e validação do alvo
 - Tarefa 1b: Identificação de substâncias ativas (*hits*)
 - Tarefa 1c: Propriedades físicoquímicas *in silico*

- **Etapa 2: De hits para protótipos**
 - Tarefa 2a: Ensaio celular da doença
 - Tarefa 2b: Relação Estrutura-Atividade
 - Tarefa 2c: Propriedades físicoquímicas *in vitro*
 - Tarefa 2d: Seletividade
 - Tarefa 2e: Genotoxicidade *in vitro*

- **Etapa 3: Otimização de protótipos**
 - Tarefa 3a: Modelo Animal da doença
 - Tarefa 3b: ADME *in vitro*
 - Tarefa 3c: ADME *in vivo*
 - Tarefa 3d: Citotoxicidade
 - Tarefa 3e: Toxicidade aguda
 - Tarefa 3f: Genotoxicidade *in vitro*
 - Tarefa 3g: Desenvolvimento farmacêutico e scale-up

- **Etapa 4: Eleição de candidato a fármaco**
 - Tarefa 4a: Excreção e identificação de metabólitos *in vivo*
 - Tarefa 4b: Distribuição tecidual *in vivo*
 - Tarefa 4c: Toxicidade subcrônica e toxicocinética
 - Tarefa 4d: Genotoxicidade *in vivo*
 - Tarefa 4e: Toxicologia reprodutiva 1
 - Tarefa 4f: Segurança *in vivo*
 - Tarefa 4g: Desenvolvimento de formulação

- **Etapa 5: Ensaio clínico de fase 1**
 - Tarefa 5a: Ensaio clínico de fase 1
- **Etapa 6: Ensaio clínico de fase 2**
 - Tarefa 6a: Ensaio clínico de fase 2
 - Tarefa 6b: Toxicidade subcrônica (3-6 meses)
- **Etapa 7: Ensaio clínico de fase 3**
 - Tarefa 7a: Ensaio clínico de fase 3
 - Tarefa 7b: Toxicologia reprodutiva 2
 - Tarefa 7c: Toxicidade crônica
 - Tarefa 7d: Carcinogenicidade

Uma vez obtidos todos os dados provenientes do processo e um fármaco viável para o tratamento de uma determinada doença, é feita uma solicitação *NDA* (*New Drug Application*) à agência reguladora para a comercialização do produto.

2.6 TRABALHOS CORRELATOS

- **G1: Quando aconteceu?:** O G1, o portal de notícias da rede Globo, oferece múltiplos jogos de *timeline* com temas variados. Nomeados *especiais*, essa série de jogos requer que o jogador posicione cartas que representam datas relacionadas a assuntos como copas do mundo, retrospectivas anuais, celebridades, eventos mundiais, entre outros (<https://g1.globo.com>);
- **NIG Brinquedos: Linha do Tempo:** O Linha do Tempo é um jogo de tabuleiro onde jogadores competem para formar linhas do tempo de fatos recentes sobre história, esportes e cultura brasileira, sendo o vencedor aquele que posicionar mais fichas corretamente no tabuleiro (<https://nigbrinquedos.com.br/>);
- **CMDF: Linha do tempo - Roma:** O Centro de Desenvolvimento de Materiais Funcionais, um centro de pesquisa apoiado pela Fundação de Amparo à Pesquisa de São Paulo, idealizou e produziu este jogo *timeline* que objetiva ordenar eventos acontecidos na Roma antiga (<https://www.ludoeducativo.com.br/pt/play/linha-do-tempo-roma>);
- **Galápagos Jogos: Timeline Clássico:** O Timeline Clássico é um jogo de cartas multijogador onde cada jogador é dado quatro cartas que devem ser posicionadas corretamente na mesa, e compra uma nova carta se cometer qualquer erro. Enquanto

o Timeline Clássico tem como tema grandes fatos históricos, a série de jogos também oferece diversos outros assuntos (<https://www.mundogalapagos.com.br/jogo-de-cartas-timeline-classic/produto/TML103>);

- **Tom J. Watson: Wikitrivia:** O Wikitrivia é um jogo *timeline* de navegador cujos fatos a serem ordenados são provenientes do Wikidata, uma base de dados irmã de projetos como Wikipedia, Wiktionary, entre outros (<https://wikitrivia.tomjwatson.com/>);
- **Javier Hernandez Torres: The Timeline Game:** Um jogo para dispositivos *iOS* onde jogadores ordenam datas de invenções, eventos e história, objetivando aprendizado e entretenimento simultâneo (<https://apps.apple.com/rs/app/the-timeline-game/id1502237831>);
- **Buffalo Games: Chronology:** Um jogo de cartas onde jogadores trocam turnos construindo linhas do tempo individuais, e o primeiro jogador a obter uma linha do tempo com 10 cartas posicionadas corretamente vence (<https://buffalogames.com/chronology/>);
- **AbacusSpiele: Anno Domini:** Uma série de jogos de cartas envolvendo blefe onde jogadores trocam turnos posicionando cartas representando eventos históricos, inicialmente sem saber suas datas. Ao posicionar uma carta, os oponentes podem ou não alegar que foi cometido um erro, sendo penalizado o jogador que posicionou a carta erroneamente se de fato ocorreu um erro, ou sendo penalizado o jogador que fez a alegação se não ocorreu (<https://abacusspiele.de/produkt-kategorie/anno-domini-serie/>).

3 ELABORAÇÃO

Esta seção tem como objetivo determinar os requisitos técnicos para o desenvolvimento da adaptação digital em modalidade timeline do SCREENER. Com este intuito, serão examinados os quesitos necessários à implementação de um jogo timeline, o comportamento do jogo a ser implementado, onde os aspectos do SCREENER original serão intercalados, e as ferramentas a serem utilizadas para estes propósitos.

3.1 PLANEJAMENTO DO JOGO

Ao ser executado, o jogo deve apresentar um menu inicial que permita ao jogador:

- **Ler informações detalhadas quanto ao assunto e regras do jogo:** É importante ao propósito de aprendizado que o jogador tenha uma ideia do que é esperado dele ao iniciar uma partida;
- **Abrir o website do SCREENER original:** É prudente mostrar aos usuários o jogo que originou o novo, tanto para contexto quanto para disseminação. Adicionalmente, o site do jogo original apresenta informações importantes quanto à inspiração e propósito do novo jogo, bem como informação mais detalhada sobre o tema de Descoberta e Desenvolvimento de Fármacos;
- **Iniciar o jogo próprio.**

O jogo próprio será constituído por 29 cartas numeradas de 1 a 29, representantes das tarefas constituintes do processo de DDF. Cada carta exibe um título, descrição, e seu número. Uma partida se inicia com uma carta aleatória na mesa, sendo visíveis seu título, descrição, e número. O jogador então aleatoriamente recebe uma das 28 cartas restantes, com título e descrição visíveis, mas número oculto. O jogador move a carta, e coloca-a antes, após, ou entre a(s) carta(s) presente(s) na mesa, na posição que acredita ser sequencialmente correta. A seguir, a jogada é avaliada. Se a carta foi posicionada corretamente, seu número é revelado e muda de cor para verde, indicando o acerto. Caso contrário, a carta é movida para a posição correta, e seu número é revelado, mudando de cor para vermelho, indicando erro. Após a avaliação da jogada, o jogador recebe uma nova carta dentre as restantes, e o processo é repetido até que todas as cartas estejam na mesa, ou que ocorram 3 erros.

Com o término do jogo, deve ser exibida uma tela listando o número de acertos, tempo decorrido, e oferecendo as opções de jogar novamente ou voltar ao menu.

3.2 REQUERIMENTOS PARA O FUNCIONAMENTO DO JOGO

Com base nas observações e definições prévias, aqui são listadas as necessidades técnicas ao funcionamento designado do jogo:

- **Informação das cartas:** A informação compreendida por cada uma das 29 tarefas do processo de DDF deve estar armazenada no jogo. Isso inclui o nome de cada etapa, as descrições destas, e também o valor sequencial de cada uma.
- **Display das cartas:** A informação (Nome, descrição, número de etapa) deve ser exibida em cartas semelhantes às do SCREENER original, de maneira que cada tarefa e aquilo que ela representa fique claramente visível.
- **Movimentação das cartas:** Deve ser possível mover as cartas exibidas de maneira arrastar-e-soltar (*drag-and-drop*), para que seja possível ordená-las de acordo com os propósitos do jogo.
- **Instanciação das cartas:** O jogo deve simular um baralho real, e portanto as cartas devem ser instanciadas aleatoriamente, de maneira que não haja repetição, até que o baralho esteja vazio ou ocorra o fim do jogo, de acordo com as regras estabelecidas.
- **Definição de áreas para cartas novas e cartas já posicionadas:** O jogo deve consistir de duas áreas principais: Uma onde são posicionadas as cartas de maneira sequencial, a "mesa", de onde cartas não podem ser retiradas; Outra de onde é possível remover cartas, a "mão", onde é instanciada uma nova carta toda vez que uma carta é posicionada na primeira área.
- **Detecção de fim do jogo:** Deve haver um sistema para detectar quando o jogo chega ao fim, seja porque todas as cartas foram posicionadas, ou porque foi cometida a quantidade máxima de erros.
- **Contador de tempo:** Deve haver um relógio que marca a duração de uma execução do jogo e exibe-a no final, para fim de avaliar o desempenho do jogador ao final de uma partida.
- **Contador de acertos e erros:** O jogo deve acompanhar quantas cartas são posicionadas corretamente e quantas não são, e exibir essas quantidades ao final de uma partida, também para fim de avaliar o desempenho do jogador.
- **Detecção de colisão entre cartas em movimento e a área "mesa":** É necessário um sistema para detectar quando uma carta sendo arrastada pelo jogador está sobre a área "mesa", para que esta possa ser posicionada e avaliada quando o jogador soltá-la.

- **Avaliação de cartas posicionadas:** É necessário um sistema que avalie a sequência de cartas sobre a mesa quando o jogador posiciona uma nova carta. Se a nova carta não foi posicionada corretamente, ela é movida para a posição correta, e um erro é considerado cometido.
- **Distinção entre cartas nas áreas mão e mesa:** Cartas na área mão devem exibir uma interrogação (caractere “?”) em vez do número de etapa, com o intuito de não permitir ao jogador saber imediatamente onde posicioná-la sem antes examinar seu conteúdo. Assim sendo, os nomes e descrições devem ser exibidos normalmente. Essa interrogação deve ser substituída pelo número de etapa real após a carta ser posicionada e avaliada, para que o jogador saiba se fez a jogada correta ou não.
- **Mudança de cor após avaliação:** Quando a posição de uma carta é avaliada, seu número de etapa deve mudar para a cor **verde** para indicar que foi posicionada corretamente, ou **vermelho** para indicar que foi cometido um erro.
- **Múltiplas telas e transição entre elas:** O jogo deve apresentar uma tela principal, uma tela para informação, e uma tela para execução do jogo próprio. Consequentemente, é necessário um sistema para mover de tela a tela.

3.3 FERRAMENTAS ESCOLHIDAS

Uma vez estabelecidos o planejamento e requerimentos, as seguintes ferramentas foram adotadas para a implementação do jogo.

- **Unity game engine:** O Unity é uma ferramenta versátil com várias vantagens, e é uma escolha coerente por diversos motivos: Permite testar a funcionalidade do jogo em tempo real e em dispositivos e telas de especificações e tamanhos diferentes, proporciona suporte robusto para gráficos 2D adequados ao jogo, e adicionalmente apresenta integração intuitiva com o Visual Studio Code. O uso de scripts C# permite uma flexibilidade valiosa para a implementação das mecânicas do jogo, e a capacidade de prototipagem rápida da *engine* permite testar ideias e refinar o jogo de maneira eficiente e rápida (<https://unity.com>). O Unity é ademais provido de um manual de uso para referência (<https://docs.unity3d.com/Manual/index.html>).
- **Visual Studio Code:** O VS Code é um editor cuja natureza leve e versátil, interface intuitiva, e integração direta com o Unity fazem dele uma escolha ideal para o desenvolvimento do jogo e a criação de scripts para as mecânicas dele (<https://code.visualstudio.com>).
- **Paint.NET:** O Paint.NET é um software de edição de imagem que apresenta suporte robusto para recursos como camadas de imagem, efeitos especiais, e uma

variedade diversa de ferramentas de edição adequadas à criação de visuais para o jogo, enquanto ainda sendo mais leve que softwares de edição de imagem alternativos (<https://www.getpaint.net>).

Determinados o comportamento do jogo, os requisitos técnicos necessários, e as ferramentas adequadas ao trabalho, parte-se subsequentemente para o desenvolvimento do jogo.

4 DESENVOLVIMENTO

Esta seção aborda o desenvolvimento e implementação do jogo de acordo com o planejamento estabelecido previamente, fazendo uso da game engine Unity, com auxílio do Visual Studio Code para *scripting* e do editor de imagem Paint.NET para elaboração de visuais.

4.1 INFORMAÇÃO DAS CARTAS

É necessário que as informações de cada carta representante das etapas do processo de DDF esteja armazenada no jogo de alguma forma, para que elas possam ser eventualmente instanciadas durante uma partida do jogo. Com este propósito, será feito uso do recurso de *Scriptable Objects* do Unity. *Scriptable Objects* são containers de informação de alta performance específicos ao Unity. Um de seus propósitos é armazenar dados imutáveis, tais como os nomes, descrições e números de etapa de cada uma das cartas. Criou-se então o template para as informações de uma carta, mediante o código 1.

Código 1 – Criação do Scriptable Object para cartas

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

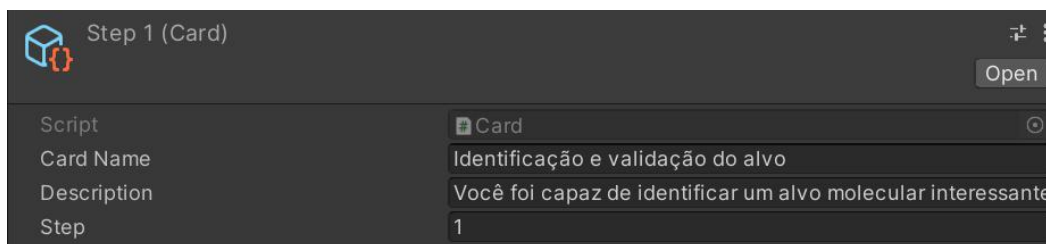
[CreateAssetMenu(fileName="Untitled Card", menuName="Card")]
public class Card : ScriptableObject
{
    public string cardName;
    public string description;
    public int step;
}
```

Com isso o editor permitirá criar objetos *Card*, cada um contendo uma string *cardName* (Título da carta), uma string *description* (Descrição da carta) e um *integer* *step* (O número de tarefa dela). Isso permite criar 29 objetos que servirão de container para as informações de cada uma das cartas, como exemplificado na figura 5.

4.2 DEFINIÇÃO DE ÁREAS PARA CARTAS

Para o funcionamento do jogo precisa-se criar uma área **mão** onde cada carta a ser posicionada pelo jogador é recebida, e uma área **mesa** onde permanecem as cartas já posicionadas. Para isso será feito uso do componente grid do Unity. *Grids* permitem alinhar objetos, neste caso as cartas do jogo, baseado num layout determinado. Todos os objetos configurados como objetos-filho do objeto com componente grid serão alinhados

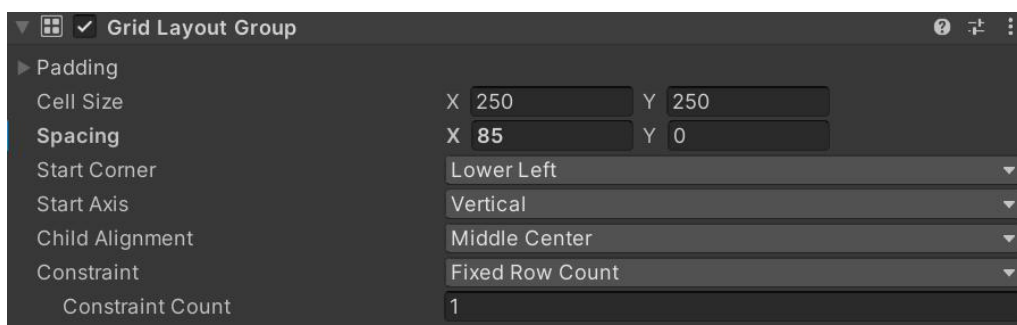
Figura 5 – Scriptable Object nomeado *Step 1*, representando a primeira tarefa do jogo



Fonte: De autoria própria, usando a Unity game engine (<https://unity.com>)

de acordo com o dito layout. Cria-se então um objeto retangular que cobre uma área horizontal da tela de jogo, representando a mesa, e atribui-se a ele um componente grid. Para os propósitos de um jogo timeline, quer-se que as cartas estejam alinhadas em sequência sobre essa área horizontal, e portanto configura-se a grid como ilustrado na figura 6.

Figura 6 – Configuração de um componente grid



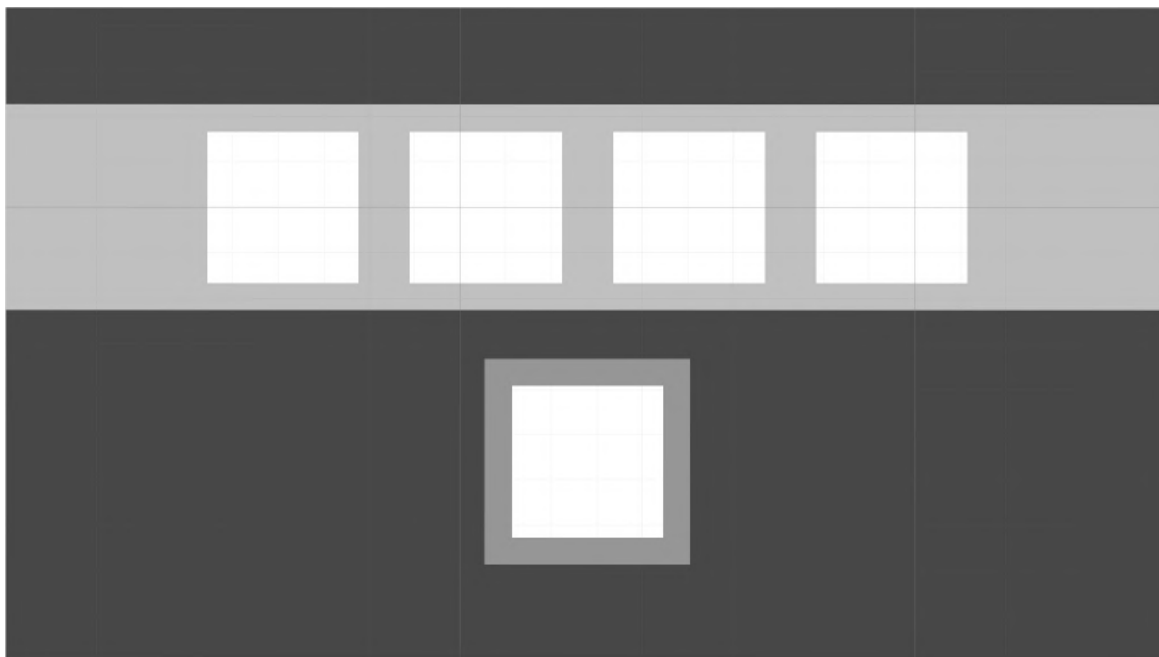
Fonte: De autoria própria, usando a Unity game engine (<https://unity.com>)

Isso resulta numa zona na qual pode-se atribuir objetos que serão automaticamente divididos em células de tamanho 250x250, espaçadas de 85 unidades. O tamanho real destas unidades do editor varia dependendo do tamanho da tela do dispositivo onde o jogo está sendo executado, mas as proporções serão constantes. Permite-se um número qualquer de colunas, mas apenas uma linha, assim simulando uma mesa onde as cartas são alinhadas numa sequência horizontal. Em seguida cria-se uma zona similar para a mão, que pode conter apenas um objeto. Por fim ficam definidas as áreas que representarão a mesa de jogo e a mão do jogador, figura 7.

4.3 MOVIMENTAÇÃO DAS CARTAS

Uma vez definidas as zonas necessárias à execução do jogo, é preciso permitir ao jogador arrastar e soltar as cartas para movimentá-las da zona *mão* à zona *mesa*. Com este intuito, criou-se o script exibido no código 2.

Figura 7 – Zona *mesa* suprida de 4 objetos-filho alinhados a sua grid, e zona *mão* suprida de 1 objeto alinhado a sua grid



Fonte: De autoria própria, usando a Unity game engine (<https://unity.com>)

Código 2 – Script para movimentação drag-and-drop de objetos

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class DragDrop : MonoBehaviour
{
    private bool isDragging = false;

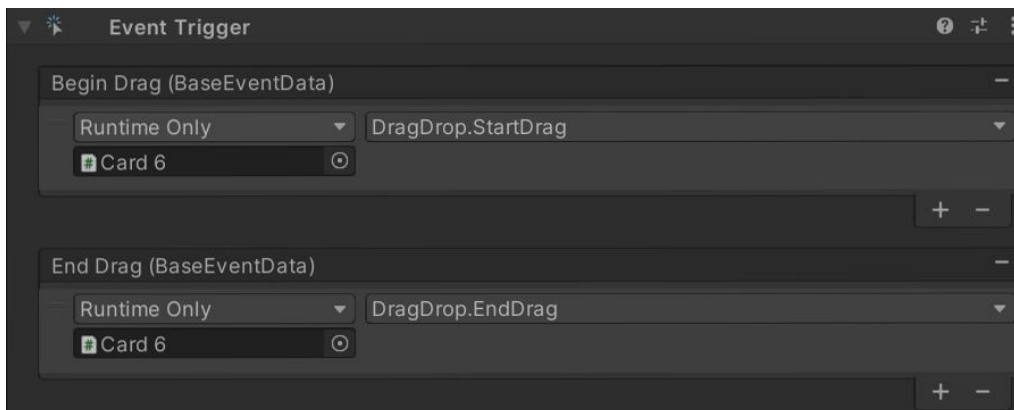
    public void StartDrag(){
        isDragging=true;
    }

    public void EndDrag(){
        isDragging=false;
    }

    void Update()
    {
        if(isDragging){
            transform.position = new Vector2(Input.mousePosition.x,
                Input.mousePosition.y);
        }
    }
}
```

Um objeto anexado do código 2 precisa então ser configurado com um componente *event trigger* de acordo com a figura 8. *Event triggers* podem ser usados para especificar funções que devem ser chamadas quando o jogador realiza uma ação.

Figura 8 – Configuração do componente *event trigger* para movimentar um objeto



Fonte: De autoria própria, usando a Unity game engine (<https://unity.com>)

Neste caso, quando o jogador começa a arrastar um objeto de jogo, a função *StartDrag()* do script é chamada, e reciprocamente, quando o objeto não está mais sendo arrastado é então chamada a função *EndDrag()*. Com isso, enquanto o objeto está sendo arrastado, é executado o conteúdo da função *Update()* do script, que é por definição da game engine chamada toda frame de jogo. Isso permite uma movimentação fluida do objeto.

4.4 DISPLAY DAS CARTAS

O conteúdo de cada carta está contido em 29 *scriptable objects* criados, mas esta informação ainda não está visível no jogo. Elabora-se então um visual para as cartas do jogo, onde essas informações serão inseridas toda vez que uma carta qualquer for instanciada. Com esse propósito, será feito uso do sistema de prefabs do Unity.

Prefabs (prefabricated) permitem criar, configurar e salvar para uso futuro um objeto de jogo complexo, incluindo componentes, propriedades, e objetos-filho. Adicionalmente, pode-se criar variações de um prefab que herdaram suas características principais mas apresentam algumas diferenças, o que se adequa ao objetivo. Deve ser criado então um prefab template representando uma carta genérica, e dela devem ser criadas variações para as 29 cartas. Para atender as necessidades do jogo, um prefab carta deve conter:

- Um plano de fundo, sobre onde o texto será exibido;
- O nome da tarefa do processo de DDF que a carta representa;
- A descrição da tarefa que a carta representa;

- O número de sequência da tarefa que a carta representa;
- A interrogação que é exibida em lugar do número de sequência da tarefa, antes que a carta seja posicionada.

Elabora-se então um objeto adequado, ilustrado na figura 9.

Figura 9 – *Prefab* elaborada para uma carta genérica



Fonte: De autoria própria, usando a Unity game engine (<https://unity.com>)

Com o template de uma carta genérica criado, é necessário estabelecer um sistema para que as informações de uma tarefa em particular, contidas nos 29 *scriptable objects*, sejam inseridas nestes objetos template quando necessário. Para isso elabora-se um script para o display adequado destas informações quando cada carta é instanciada no jogo, exibido no código 3.

Código 3 – Script para display das informações das cartas

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using TMPro;

public class CardDisplay : MonoBehaviour
{
    public Card card;
    public TextMeshProUGUI cardName;
    public TextMeshProUGUI cardDescription;
    public TextMeshProUGUI cardStep;

    void Start()
    {
        cardName.text=card.cardName;
        cardDescription.text=card.description;
        cardStep.text=card.step.ToString();
    }
}

```

A função *Start()* é, por definição da game engine, executada exatamente uma vez, no momento que um objeto acoplado do script é instanciado. No script exibido no código 3, ela tem como função substituir o texto *placeholder* do objeto *template* pelo texto real de uma das 29 tarefas. Com este intuito, cria-se então 29 variações do *prefab template*, cada uma acoplada dos scripts de movimentação *drag-and-drop* e *display*.

4.5 DETECÇÃO DE COLISÃO ENTRE CARTAS E ÁREAS

Agora que há cartas visualmente distintas, e ademais está estabelecido um sistema para movimentá-las, é necessário estabelecer um sistema para que, quando uma carta arrastada sobre a zona *mesa* seja solta, ela alinhe-se à grid da zona. Em outras palavras, é preciso detectar quando uma carta está em movimento sobre uma zona, e quando não está. Para isso, o sistema de colisão entre objetos proporcionado pelo Unity é de interesse.

Quando objetos acoplados de componentes collider interagem, ou mais especificamente neste caso, entram em contato, o sistema de *scripting* é capaz de identificar quando um collider entra no espaço do outro e permite realizar diversas ações por meio do uso da função *OnCollisionEnter()*. Conversamente, usa-se a função *OnCollisionExit()* quando dois objetos não estão mais em contato. Pode-se fazer uso desse sistema para elaborar uma modificação ao script de movimento do código 2. Exibe-se esse script modificado no código 4.

Assim, qualquer objeto carta acoplado do script de movimentação modificado, e su-

prido também de um componente collider, irá ser adequadamente alinhado à grid da zona *mesa* quando o jogador arrastá-lo e soltá-lo sobre ela, por meio da função *SetParent()*. Caso não esteja sobre a zona *mesa*, simplesmente voltará para sua localização original, a zona *mão*.

Como é somente necessário detectarmos cartas em colisão com zonas, e não cartas em colisão com cartas ou zonas em colisão com zonas, para evitar quaisquer problemas, altera-se a matriz de colisão do projeto para que colisões carta-carta e zona-zona sejam ignoradas, como ilustrado na figura 10.

Figura 10 – Matriz de colisão do projeto

	Default	TransparentFX	Ignore Raycast	Water	UI	Cards	Zones
Default	✓	✓	✓	✓	✓	✓	✓
TransparentFX	✓	✓	✓	✓	✓	✓	✓
Ignore Raycast	✓	✓	✓	✓	✓	✓	✓
Water	✓	✓	✓	✓	✓	✓	✓
UI	✓	✓	✓	✓	✓	✓	✓
Cards	✓	✓	✓	✓	✓	✓	✓
Zones	✓	✓	✓	✓	✓	✓	✓

Fonte: De autoria própria, usando a Unity game engine (<https://unity.com>)

Código 4 – Script de movimentação com adição de sistema para detectar colisões e alinhar cartas à zona *mesa*

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class DragDrop : MonoBehaviour
{
    public GameObject Canvas;
    public GameObject Zone;
    private bool isDragging = false;
    private bool isWithinZone = false;
    private GameObject startParent;
    private Vector2 startPosition;
    void Start(){
        Canvas = GameObject.Find("Canvas");
        Zone = GameObject.Find("DropZone");
    }
    private void OnCollisionEnter2D(Collision2D collision){
        isWithinZone=true;
        Zone=collision.gameObject; }
    private void OnCollisionExit2D(Collision2D collision){
        isWithinZone=false;
        Zone=null; }
    public void StartDrag(){
        if(transform.parent.name!="DropZone"){
            isDragging=true;
            startParent = transform.parent.gameObject;
            startPosition = transform.position;
        }
    }
    public void EndDrag(){
        isDragging=false;
        if(isWithinZone){
            transform.SetParent(Zone.transform, false);
        }
        else{
            transform.SetParent(startParent.transform, false);
            transform.position=startPosition;
        }
    }
    void Update(){
        if(isDragging){
            transform.position = new Vector2(Input.mousePosition.x,
                Input.mousePosition.y);
            transform.SetParent(Canvas.transform, true);
        }
    }
}

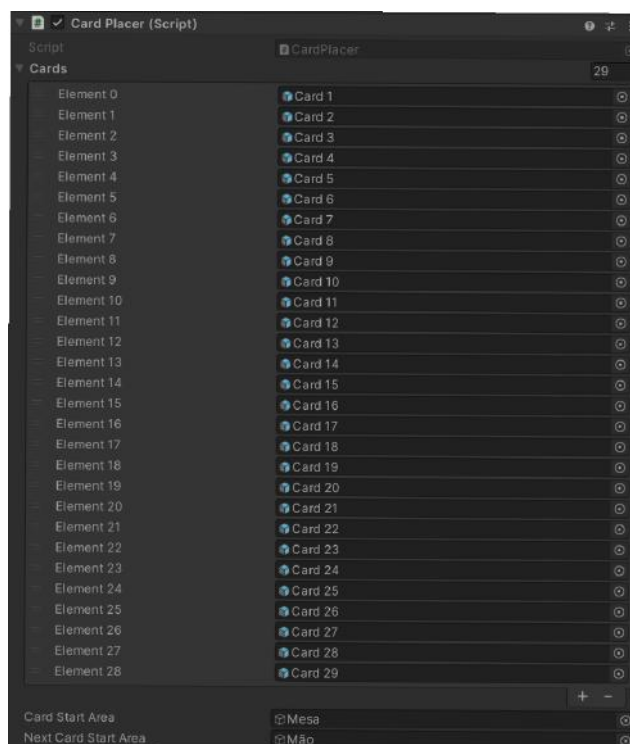
```


4.6 INSTANCIACÃO DAS CARTAS

O funcionamento do jogo requer que, ao começo de uma partida, uma carta aleatória seja instanciada na zona *mão*. Em seguida, cada vez que uma carta for posicionada na zona *mesa*, uma nova carta dentre as restantes no baralho deve ser instanciada na zona *mão*, assim simulando o baralho sendo esvaziado. Esse comportamento requer que o conjunto de 29 cartas seja embaralhado antes da execução de cada rodada do jogo. Adicionalmente, é também necessário que saiba-se quando uma nova carta deve ser dada ao jogador, ou em outras palavras, quando uma carta é posicionada na mesa. Soluciona-se essa questão com o script de jogo exibido no código 5.

As 3 variáveis públicas do código 5, sendo estas a lista *cards*, objeto de jogo *cardStartArea* e objeto de jogo *nextCardStartArea*, respectivamente recebem as 29 *prefabs* de cartas criadas, a zona *mesa*, e a zona *mão*, como ilustrado na figura 11.

Figura 11 – Variáveis públicas do script de jogo



Fonte: De autoria própria, usando a Unity game engine (<https://unity.com>)

Inicialmente, com a função *Start()*, o script do código 5 embaralha de maneira aleatória a lista que representa as 29 cartas de jogo. Em seguida, ele instancia uma carta alinhada à grid da zona *mesa*, representada pelo *GameObject* nomeado *cardStartArea*, e então instancia uma carta alinhada à grid da zona *mão*, representada pelo *GameObject* nomeado *nextCardStartArea*, para que o jogo possa ser começado. Com a função *Update()*, toda vez que o jogador posicionar uma carta na zona *mesa*, uma nova carta será instanciada na zona *mão*, até que por fim as 29 cartas do baralho sejam posicionadas.

Código 5 – Script de jogo com instanciação adequada das cartas

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class CardPlacer : MonoBehaviour
{
    public List<GameObject> cards;
    public GameObject cardStartArea;
    public GameObject nextCardStartArea;
    private int i = 0;
    private List<int> seed = new List<int> { 0, 1, 2, 3, 4, 5, 6, 7, 8,
        9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24,
        25, 26, 27, 28 };

    void Start(){
        for(int x = 0; x < seed.Count; x++){
            int temp = seed[x];
            int randomIndex = Random.Range(x, seed.Count);
            seed[x] = seed[randomIndex];
            seed[randomIndex] = temp;
        }
        if(i==0){
            GameObject card = Instantiate(cards[seed[i]], new Vector2
                (0,0), Quaternion.identity);
            card.transform.SetParent(cardStartArea.transform,false);

            i+=1;
            card = Instantiate(cards[seed[i]], new Vector2(0,0),
                Quaternion.identity);
            card.transform.SetParent(nextCardStartArea.transform,false);
        }
    }

    void Update(){
        if(cardStartArea.transform.childCount==(i+1) ){
            i+=1;
            if(i<29){
                GameObject card = Instantiate(cards[seed[i]], new
                    Vector2(0,0), Quaternion.identity);
                card.transform.SetParent(nextCardStartArea.transform,
                    false);
            }
        }
    }
}

```

4.7 AVALIAÇÃO DE CARTAS POSICIONADAS E SUBSEQUENTE MUDANÇA DE COR

Um jogo timeline deve avaliar cada carta posicionada sobre a mesa. Se uma carta for posicionada erroneamente, ela deve ser automaticamente movida para a posição correta, e o jogador deve ser informado do erro cometido. Conversamente, se uma carta for posicionada corretamente, o jogador deve ser informado do acerto. Em consequência disso, será necessário estabelecer um sistema que avalia a corretude de uma carta recém-posicionada na zona *mão*, reordena as cartas alinhadas à grid da zona se ocorreu erro, e informa o jogador do resultado. Para esse propósito, elaboram-se modificações ao script de jogo do código 5.

A função *Start()* e a função *Update()* do script modificado são exibidas, respectivamente, nos códigos 6 e 8. Além das variáveis já presentes no código 5 original, estas funções fazem uso de duas novas variáveis: Uma lista nomeada *checklist*, usada junto da função exibida no código 7 para avaliar a corretude da sequência de cartas na mesa, e um *GameObject* nomeado *lastCardTouched*, usado para demarcar a carta que está sendo avaliada.

Código 6 – Função *Start()* do script de jogo modificado para avaliação de cartas

```
void Start(){
    for(int x = 0; x < seed.Count; x++){
        int temp = seed[x];
        int randomIndex = Random.Range(x, seed.Count);
        seed[x] = seed[randomIndex];
        seed[randomIndex] = temp;
    }
    if(i==0){
        GameObject card = Instantiate(cards[seed[i]], new
            Vector2(0,0), Quaternion.identity);
        card.transform.GetChild(0).GetComponent<Image>().color =
            new Color32(0,255,0,255);
        card.transform.SetParent(cardStartArea.transform,false);
        card.transform.GetChild(4).gameObject.SetActive(false);
        i+=1;
        card = Instantiate(cards[seed[i]], new Vector2(0,0),
            Quaternion.identity);
        card.transform.SetParent(nextCardStartArea.transform,
            false);
        lastCardTouched=card;
    }
}
```

Código 7 – Função de avaliação de cartas posicionadas

```
private static bool IsSorted(List<int> Test){
    for (int x = 1; x < Test.Count; x++){
        if (Test[x - 1] > Test[x]){
            return false;
        }
    }
    return true;
}
```

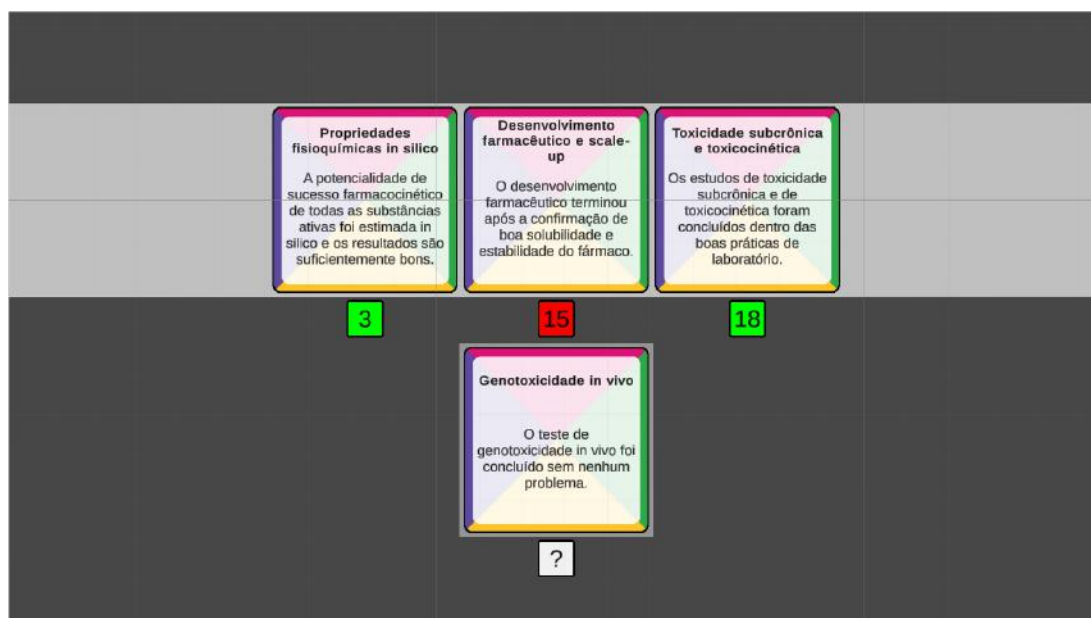
Código 8 – Função *Update()* do script de jogo modificado para avaliação de cartas

```
void Update(){
    if(cardStartArea.transform.childCount==(i+1) ){
        for(int x = 0; x<cardStartArea.transform.childCount; x
            ++){
            checklist.Add(int.Parse(cardStartArea.transform.
                GetChild(x).GetChild(3).GetComponent<TMP_Text>
                >().text));
        }
        if(IsSorted(checklist)){
            lastCardTouched.transform.GetChild(0).
                GetComponent<Image>().color = new Color32
                (0,255,0,255);
        }
        else{
            lastCardTouched.transform.GetChild(0).
                GetComponent<Image>().color = new Color32
                (255,0,0,255);
            checklist.Sort();
            lastCardTouched.transform.SetSiblingIndex(
                checklist.FindIndex(a => a == int.Parse(
                lastCardTouched.transform.GetChild(3).
                GetComponent<TMP_Text>().text)));
        }
        i+=1;
        if(i<29){
            GameObject card = Instantiate(cards[seed[i]],
                new Vector2(0,0), Quaternion.identity);
            card.transform.SetParent(nextCardStartArea.
                transform,false);
            lastCardTouched=card;
        }
    }
    checklist.Clear();
}
```

Todo objeto carta tem um sub-objeto representado por uma interrogação, que cobre seu número de tarefa. Quando uma carta é alinhada à zona *mesa*, esse objeto é desativado com a função *SetActive()*, tornando visível o número de tarefa, e a cor do dito número de tarefa é alterada para verde ou vermelho de acordo com a avaliação da jogada. Essa avaliação é feita por meio da lista *checklist*, que é preenchida com a atual sequência de cartas sobre a mesa a cada jogada e então avaliada com a função do código 7, que checa se a lista está ordenada sequencialmente. Se uma carta é posicionada erroneamente, usa-se a função *SetSiblingIndex* para movê-la para a posição correta na grid da zona.

A figura 12 ilustra uma partida em progresso, iniciada com a carta de número de tarefa 18 sobre a mesa, onde o jogador posicionou a carta de número 3 corretamente e a carta de número 15 erroneamente. A carta de número de tarefa 19, *Genotoxicidade in vivo*, tem seu número oculto, pois não foi ainda posicionada.

Figura 12 – Partida em progresso



Fonte: De autoria própria, usando a Unity game engine (<https://unity.com>)

4.8 DETECÇÃO DE FIM DE JOGO E CONTADOR DE ERROS

Uma vez estabelecido um sistema para avaliação de cartas posicionadas, é necessário elaborar um sistema para interromper o jogo quando ele chega ao fim. O jogo deverá chegar ao fim de uma partida quando uma das seguintes condições é satisfeita:

- Todas as 29 cartas são posicionadas sem que o valor máximo de três erros seja atingido;
- O valor máximo de três erros é atingido.

Com estas condições em mente, estabelece-se um contador de erros na função *Update()* do script de jogo, por meio do uso de uma variável inteiro *MistakeCount*, inicialmente atribuída de 0 e incrementada de 1 a cada erro, e uma função vazia *GameOver()*, executada quando ocorre o fim de uma partida. As modificações feitas à função *Update()* do script de jogo são exibidas no código 9.

Código 9 – Função *Update()* do script de jogo modificado

```

void Update () {
    if (cardStartArea.transform.childCount==(i+1) ) {
        for (int x = 0; x < cardStartArea.transform.childCount; x
            ++){
            checklist.Add(int.Parse(cardStartArea.transform.
                GetChild(x).GetChild(3).GetComponent<TMP_Text>().text));
        }
        if (IsSorted(checklist)) {
            lastCardTouched.transform.GetChild(0).
                GetComponent<Image>().color = new Color32
                (0,255,0,255);
        }
        else {
            lastCardTouched.transform.GetChild(0).
                GetComponent<Image>().color = new Color32
                (255,0,0,255);
            checklist.Sort();
            lastCardTouched.transform.SetSiblingIndex(
                checklist.FindIndex(a => a == int.Parse(
                    lastCardTouched.transform.GetChild(3).
                    GetComponent<TMP_Text>().text)));
            MistakeCount+=1;
        }
        i+=1;
        if (i < 29 && MistakeCount < 3) {
            GameObject card = Instantiate(cards[seed[i]],
                new Vector2(0,0), Quaternion.identity);
            card.transform.SetParent(nextCardStartArea.
                transform,false);
            lastCardTouched=card;
        }
    }
    checklist.Clear();
    if (i==29 || MistakeCount >= 3) {GameOver();}
}

```

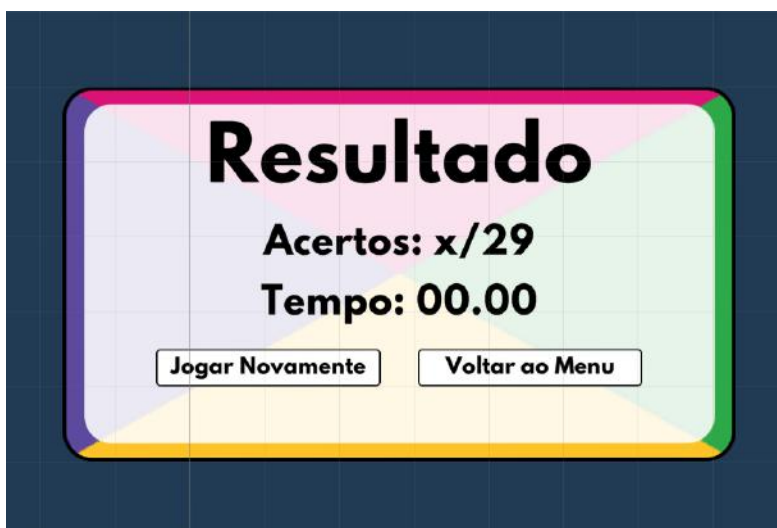
4.9 FIM DE JOGO

O script de jogo agora contém uma função *GameOver()* chamada quando ocorre o fim de uma partida, como exibido no código 9. Quando ocorre o fim de jogo, é necessário que seja exibida uma janela mostrando se ocorreu vitória ou derrota, e também a quantidade de erros cometidos. No todo, a janela deve apresentar os seguintes componentes:

- Um plano de fundo sobre o qual as informações são exibidas;
- A pontuação obtida, exibida em termos de acertos e erros;
- O tempo decorrido desde o início da partida;
- Botões para iniciar uma nova partida e voltar ao menu principal.

Elabora-se um prefab para um objeto de jogo que satisfaz estes requerimentos, ilustrado na figura 13.

Figura 13 – Janela de fim de jogo



Fonte: De autoria própria, usando a Unity game engine (<https://unity.com>)

Uma vez criado este template para a janela de fim de jogo, em seguida é necessário elaborar o conteúdo da função *GameOver()* que é chamada quando ocorre um terceiro erro, de forma que as informações adequadas sejam exibidas na janela quando ela for mostrada. O código 10 exhibe a função elaborada. O funcionamento baseia-se em substituir o texto placeholder do prefab com as informações da partida atual e fazer uso da função *SetActive()* para exibir a janela de fim de jogo, representada pelo objeto nomeado *GameOverPopUp* e inicialmente inativa.

Código 10 – Função de fim de jogo *GameOver()*

```

void GameOver(){
    GameOverPopUp.SetActive(true);
    if(MistakeCount >=3){
        GameOverPopUp.transform.GetChild(0).GetComponent<
            TMP_Text>().text="Derrota";
    }
    else{
        GameOverPopUp.transform.GetChild(0).GetComponent<
            TMP_Text>().text="Sucesso";
    }
    GameOverPopUp.transform.GetChild(1).GetComponent<TMP_Text>().
        text = ( "Acertos: " + (i-MistakeCount).ToString() + "/29" );
    GameOverPopUp.transform.GetChild(2).GetComponent<TMP_Text>().
        text = ( "Tempo: " + Stopwatch.GetComponent<Timer>().retTime
            () + "s" );
}

```

4.10 CONTADOR DE TEMPO

A função de fim de jogo exibida no código 10 faz uso de um objeto de jogo nomeado *StopWatch* para exibir a duração da partida, mas este objeto ainda não foi criado. Cria-se então um objeto cujo único componente é um script contador de tempo, cujo propósito único é medir a duração de uma partida. O script contador de tempo é exibido no código 11. Ele faz uso da classe *Time* do Unity para medir o intervalo de tempo em segundos entre cada frame de jogo, ou *deltaTime*, e soma estes intervalos na variável *clock*.

Código 11 – Script contador de tempo

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class Timer : MonoBehaviour
{
    float clock;
    bool running;
    void Start(){
        clock=0;
        running=false;
    }
    void Update(){
        if(running){clock += Time.deltaTime;}
    }
    public void startbutton(){
        running=true;
    }
    public void stopbutton(){
        running=false;
    }
    public string retTime(){
        return System.Math.Round(clock,2).ToString();
    }
}

```

4.11 SLIDER PARA CARTAS SOBRE A MESA

O baralho do jogo é compreendido por 29 cartas e conseqüentemente, conforme uma partida avança, não é possível exibir na tela de um dispositivo todas as cartas posicionadas sobre a mesa simultaneamente. Para solucionar esse problema, implementa-se um slider cujo propósito é permitir ao jogador mover a zona *mesa* em direção horizontal. O script associado a esse slider é adicionado como componente do objeto representante da zona *mesa*, e é exibido no código 12. A utilidade do slider é ilustrada na figura 14, que mostra uma partida em progresso com 7 cartas já sobre a mesa, estando algumas destas cartas ocultas à esquerda, fora do campo de visão da tela por meio de uso do slider.

Código 12 – Script para movimentação da zona *mesa* via slider

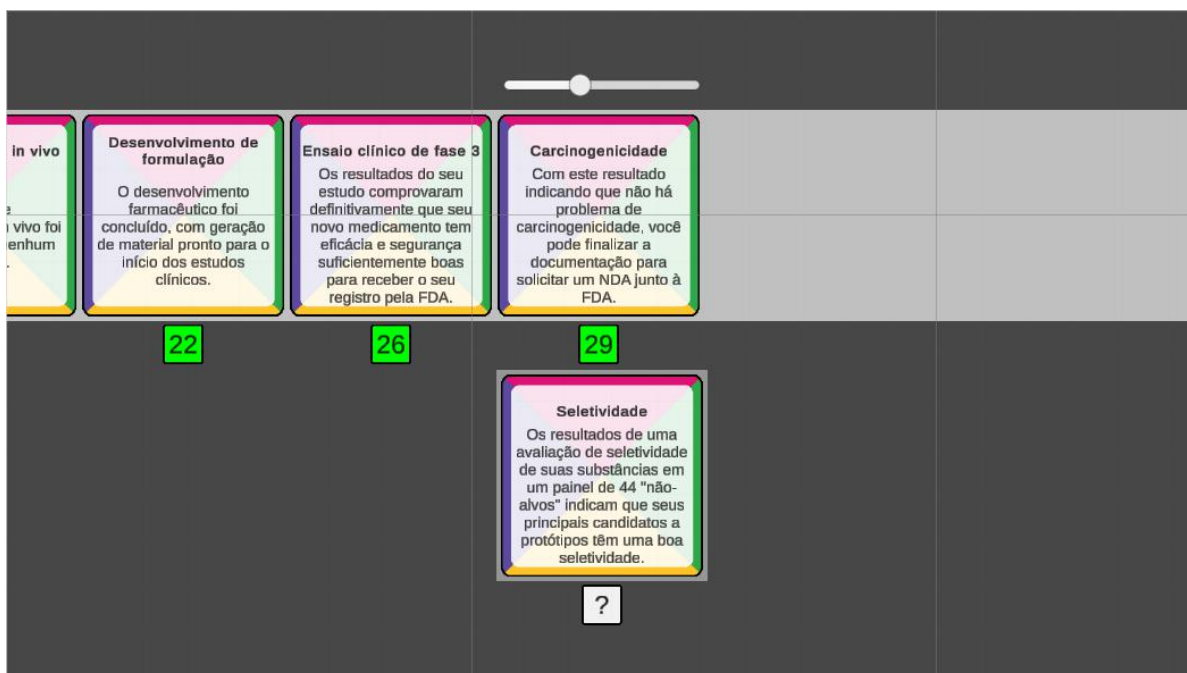
```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class DropZoneMover : MonoBehaviour
{
    public Slider DZSlider;
    private Vector3 pos;
    private Vector3 org;
    void Start()
    {
        org = this.transform.localPosition;
        pos = this.transform.localPosition;
    }
    void Update()
    {
        pos.x=org.x+(DZSlider.value-0.5f)*8080;
        this.transform.localPosition=pos;
    }
}

```

Figura 14 – Partida em progresso, com utilização de slider



Fonte: De autoria própria, usando a Unity game engine (<https://unity.com>)

4.12 TRANSIÇÃO ENTRE TELAS

O jogo deve apresentar três telas principais:

- A tela de jogo, onde ocorrem as partidas do jogo;
- Um menu principal, apresentando botões para iniciar uma partida, ler mais informações, e acessar o site do SCREENER original;
- Uma tela de informações que explica o funcionamento, propósito e tema do jogo.

Com este intuito, o sistema de scenes, ou cenas, do Unity é de interesse. Cenas são recursos que contêm todo ou parte do conteúdo de um projeto. Para os propósitos do jogo, cria-se três cenas para representar cada tela principal.

É adicionalmente necessário elaborar um script para que ocorra uma transição entre essas telas quando o botão adequado é pressionado. O script de transição entre telas é exibido no código 13. Ele é acoplado a um objeto de jogo vazio presente em cada uma das cenas, e os botões presentes nestas fazem uso das funções *ChangeScene()* e *ResetScene()* para transicionar o jogo para uma tela diferente ou reiniciar a tela atual.

Código 13 – Script para transição de telas

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

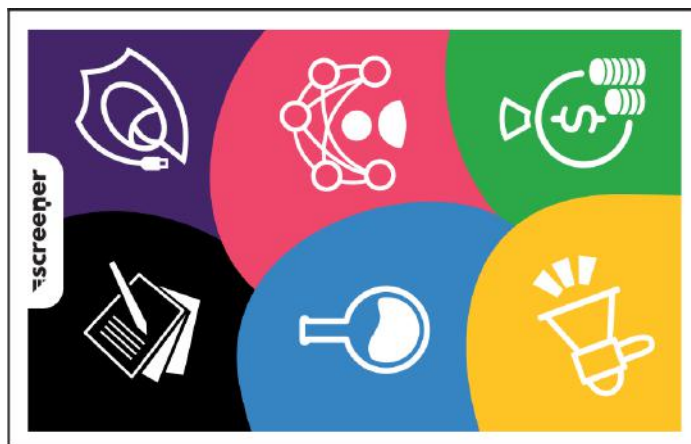
public class SceneChanger : MonoBehaviour
{
    public void ChangeScene(string sceneName){
        SceneManager.LoadScene(sceneName);
    }
    public void ResetScene(){
        SceneManager.LoadScene(SceneManager.GetActiveScene().
            name);
    }
    public void Exit(){
        Application.Quit();
    }
}
```

4.13 DESIGN VISUAL

Para a apresentação visual do jogo, fez-se uso do esquema de cores e das ilustrações do SCREENER original para gerar visuais semelhantes que servem as necessidades do novo jogo. Exemplifica-se esse princípio com a figura 15, que mostra o verso de uma carta do

SCREENER original, e a figura 16, que mostra o plano de fundo do jogo criado, feito a partir da edição da ilustração da carta.

Figura 15 – Carta de poder do SCREENER



Fonte: Materiais para Download do SCREENER, Cartas - Coloridas (screener.com.br/#download)

Figura 16 – Plano de fundo do jogo criado, adaptado do jogo original



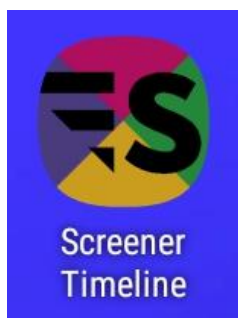
Fonte: Criado por meio do editor de imagens Paint.NET (<https://www.getpaint.net/>) com base no design visual do SCREENER original

4.14 RESULTADO: SCREENER TIMELINE

O jogo implementado, nomeado *Screener Timeline*, é um aplicativo para dispositivos *android*, como ilustrado na figura 17.

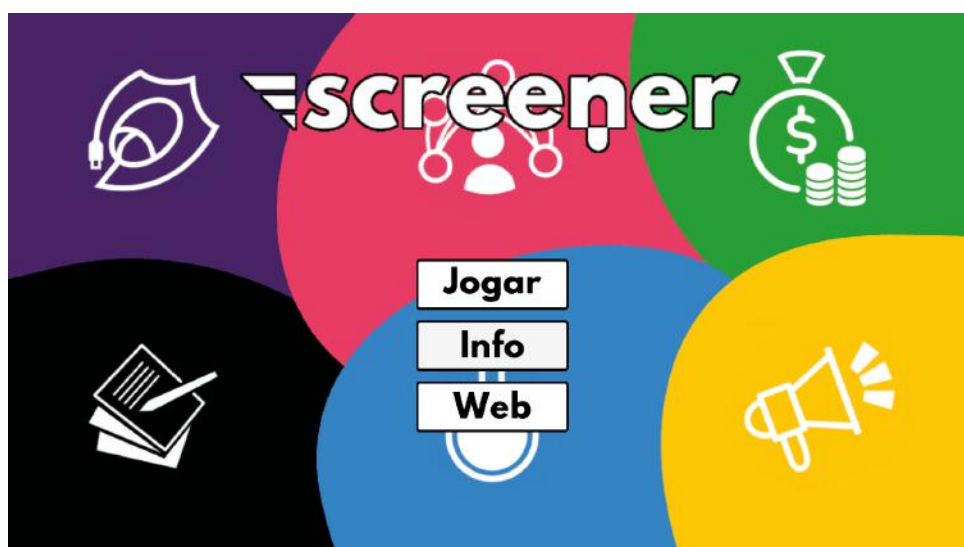
Ao iniciar o jogo, o jogador se encontra na tela do menu principal, exibida na figura 18.

Figura 17 – Ícone do jogo em ambiente android



Fonte: De autoria própria, captura de tela de um dispositivo android

Figura 18 – Menu principal do Screener Timeline



Fonte: De autoria própria, captura de tela de um dispositivo android

No menu principal, os botões *Jogar*, *Info* e *Web* respectivamente levam o jogador à tela de jogo, à tela de informações, e ao website do SCREENER original.

A tela de informações é exibida na figura 19. Ela apresenta informações quanto às regras, origem e tema do jogo, e também descreve o que o jogador pode esperar dele. O botão *Voltar* retorna o jogador ao menu principal.

A tela de jogo é exibida na figura 20. É nesta tela que ocorrem as partidas do jogo, ao fim das quais a janela de fim de jogo, exibida na figura 21, é mostrada.

A janela de fim de jogo exibe o número de acertos e a duração da partida. O botão *Jogar Novamente* reinicia a tela de jogo, e o botão *Voltar ao Menu* leva o jogador de volta ao menu principal.

Com isso, obtém-se uma adaptação digital adequada do processo de DDF compreendido pelo SCREENER em formato timeline.

Figura 19 – Tela de informações do Screener Timeline



Fonte: De autoria própria, captura de tela de um dispositivo android

Figura 20 – Tela de jogo do Screener Timeline



Fonte: De autoria própria, captura de tela de um dispositivo android

Figura 21 – Término de uma partida do Screener Timeline



Fonte: De autoria própria, captura de tela de um dispositivo android

5 AVALIAÇÃO

5.1 QUESTIONÁRIO

O Screener Timeline foi publicado no website de hospedagem de jogos *itch.io*, uma plataforma gratuita e aberta que proporciona a distribuição e download de jogos independentes, ilustrada na figura 22.

Figura 22 – Página de download do Screener Timeline

Screener Timeline

Resumo

O SCREENER (<https://www.screener.com.br/>) é um jogo educacional para o ensino do processo de Descoberta e Desenvolvimento de Fármacos e medicamentos (DDF). Normalmente um jogo de tabuleiro para até seis jogadores, o Screener Timeline é uma versão condensada dele para dispositivos Android e para um único jogador, visando promover o mesmo aprendizado lúdico e interativo de maneira compacta.

O jogo é composto por diversas cartas, cada uma representando uma das etapas do processo de DDF. Uma rodada começa com uma carta qualquer no centro da mesa, e outra carta sendo dada ao jogador. O jogador então coloca sua carta na mesa, na posição que ele acredita ser adequada à etapa (após ou anterior às cartas já presentes), e é então dado uma nova carta.

Tente posicionar todas as 29 cartas em ordem sem cometer erros, assim formando a linha do tempo do processo de fármacos.

Informação

Trabalho realizado como parte do projeto de conclusão de curso de Ciência da Computação na Universidade Federal do Rio de Janeiro.


Desenvolvido por:

Hugo Bevilaqua de Carvalho Reis

[More information](#) ▾

Download

[Download](#) Screener Timeline APK 23 MB



Fonte: <https://hugobevilaqua.itch.io/screener-timeline>

Com o intuito de avaliar o projeto, fez-se um levantamento de opiniões e feedback por meio de um questionário aplicado a usuários potenciais do jogo, com base num modelo de reação a jogos educacionais criado por Geraldo Xexéo e Alexandre Vaz (VAZ; XEXEO, 2022). O questionário e as 13 respostas coletadas são exibidos respectivamente nos apêndices B e C. Através dos resultados obtidos, observa-se que o ponto de maior dificuldade encontrado por usuários foi o nível de dificuldade do jogo, seguido por dificuldades em entender o objetivo do jogo. Ocorreram também relatos de dificuldades ao posicionar as cartas.

5.2 MELHORIAS

Jogadores expressaram dificuldade em ordenar o grande número de cartas do jogo corretamente. Adicionalmente, embora o SCREENER represente o processo de DDF em

etapas e tarefas sequenciais, em um ambiente real algumas destas etapas podem ocorrer simultaneamente ou em ordem diferente. Com isso em mente, o jogo foi alterado de forma que alguns grupos particulares de cartas possuam o mesmo número de tarefa, e portanto possam ser posicionadas adjacentes na mesa em qualquer ordem. Isso resultou na seguinte alteração aos números de tarefa de cada uma das 29 cartas de jogo:

- 1: Identificação e validação do alvo
- 2: Identificação de substâncias ativas (*hits*)
- 3: Propriedades fisicoquímicas *in silico*
- 4: Ensaio celular da doença
- 5: Relação Estrutura-Atividade
- 6: Propriedades fisicoquímicas *in vitro*
- 6: Seletividade
- 6: Genotoxicidade *in vitro*
- 7: ADME *in vitro*
- 7: Citotoxicidade
- 8: Modelo Animal da doença
- 8: ADME *in vivo*
- 8: Genotoxicidade *in vitro*
- 9: Toxicidade aguda
- 10: Desenvolvimento farmacêutico e scale-up
- 11: Excreção e identificação de metabólitos *in vivo*
- 11: Distribuição tecidual *in vivo*
- 11: Toxicidade subcrônica e toxicocinética
- 12: Genotoxicidade *in vivo*
- 12: Segurança *in vivo*
- 13: Toxicologia reprodutiva 1
- 13: Desenvolvimento de formulação

- 14: Ensaio clínico de fase 1
- 15: Ensaio clínico de fase 2
- 15: Toxicidade subcrônica (3-6 meses)
- 16: Ensaio clínico de fase 3
- 16: Toxicologia reprodutiva 2
- 16: Toxicidade crônica
- 17: Carcinogenicidade

Cartas que compartilham o mesmo número de tarefa podem ser posicionadas em qualquer ordem, dado que esta posição seja após cartas de número menor e anterior a cartas de número maior. Em outras palavras, as 29 cartas devem ser ordenadas de maneira não decrescente.

Adicionalmente, foi expressada dificuldade mecânica em posicionar uma carta entre duas outras. Para remediar isso, a área de colisão de cada carta foi moderadamente diminuída.

Por fim, com o intuito de solucionar o problema de entendimento do objetivo do jogo, o texto contido na tela de informações foi reescrito para mais claramente explicar as regras e objetivo do jogo.

6 CONCLUSÃO

Em consequência dos atributos particulares apresentados por jogos educacionais e jogos no formato timeline, o Screener Timeline apresenta diversas características de interesse a seu propósito educacional. O ato de posicionar as cartas em sequência requer que o jogador avalie o conteúdo de cada uma delas, o que significa que simultaneamente ocorre aprendizagem e memorização dos eventos constituintes do processo de Descoberta e Desenvolvimento de Fármacos, e assim são também treinadas as habilidades de análise e inferência. É proporcionado um tipo de aprendizado distinto ao do SCREENER original, dado que o novo jogo é de jogador único enquanto o original é de multijogador, e ao organizar as cartas de jogo em uma representação de linha do tempo, o jogador elucida o contexto entre cada uma das etapas e tarefas do processo de DDF, e é também proporcionado de uma oportunidade para recriar e relembrar informação aprendida anteriormente. Adicionalmente, os contadores de erros e de tempo permitem ao jogador avaliar seu desempenho ao final de uma partida, e é provido fácil acesso ao website do SCREENER caso sejam desejadas informações mais detalhadas quanto à inspiração do jogo ou o tema de DDF.

A representação visual simples do jogo, em conjunto com a duração curta de cada partida individual e formato compacto, fazem do Screener Timeline uma ferramenta útil para o auxílio da aprendizagem do tema de Descoberta e Desenvolvimento de Fármacos.

REFERÊNCIAS

BACKLUND, P.; HENDRIX, M. Educational games – are they worth the effort? a literature survey of the effectiveness of serious games. **IEEE Xplore**, United Kingdom, 2013. Disponível em: <https://ieeexplore.ieee.org/document/6624226>. Acesso em: 2 set.2023.

BERTILSSON, F. et al. Retrieval practice: Beneficial for all students or moderated by individual differences? **Psychology Learning and Teaching**, Sweden, v. 20, n. 1, 2020. Disponível em: <https://journals.sagepub.com/doi/10.1177/1475725720973494>. Acesso em: 22 jul.2023.

BUTLER, S. **Chronological Understanding**. In: **Debates in History Teaching**. Gloucestershire: Routledge, 2017. 12 p.

KALMPOURTZIS, G. **Educational Game Design Fundamentals: A Journey to Creating Intrinsically Motivating Learning Experiences**. Boca Raton, Florida: CRC Press, 2018. 360 p.

KARBALAEI, A. Critical thinking and academic achievement. **Íkala, Revista de Lenguaje y Cultura**, Medellín, v. 17, n. 2, 2012. Disponível em: http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0123-34322012000200001&lng=en&nrm=iso. Acesso em: 23 jul.2023.

LOUNSBURY, N. et al. Creation and implementation of a drug discovery and development game. **Currents in Pharmacy Teaching and Learning**, Larkin University College of Pharmacy, Department of Pharmaceutical Sciences, 18301 N Miami Ave, Miami, FL 33169, United States, v. 14, n. 2, 2022. Disponível em: <https://www.sciencedirect.com/science/article/abs/pii/S187712972100349X?via%3Dihub>. Acesso em: 2 set.2023.

NOEL, F.; XEXEO, G. **Screeener: Jogo educacional para o ensino do processo de descoberta e desenvolvimento de fármacos**. Rio de Janeiro: [s.n.], 2021. 120 p.

NOEL, F. et al. Screeener, an educational game for teaching the drug discovery and development process. **Brazilian Journal of Medical and Biological Research**, Rio de Janeiro, v. 54, n. 12, 2021. Disponível em: <https://www.bjournal.org/article/screeener-an-educational-game-for-teaching-the-drug-discovery-and-development-process>. Acesso em: 20 jul.2023.

OESTREICH, J. H.; GUY, J. W. Game-based learning in pharmacy education. **Pharmacy**, Department of Pharmaceutical Sciences, College of Pharmacy, University of Findlay, Findlay, OH 45840, USA, v. 10, n. 1, 2022. Disponível em: <https://www.sciencedirect.com/science/article/abs/pii/S187712972100349X?via%3Dihub>. Acesso em: 2 set.2023.

SILVA, S. L. D.; CATARCI, T. Visualization of linear time-oriented data: a survey. **IEEE Xplore**, Hong Kong, 2000. Disponível em: <https://ieeexplore.ieee.org/document/882407?arnumber=882407>. Acesso em: 23 jul.2023.

VAZ, A.; XEXEO, G. Desenvolvimento de um modelo de reação a jogos educacionais digitais. In: **Anais Estendidos do XXI Simpósio Brasileiro de Jogos e Entretenimento Digital**. Porto Alegre, RS, Brasil: SBC, 2022. p. 623–632. ISSN 0000-0000. Disponível em: https://sol.sbc.org.br/index.php/sbgames_estendido/article/view/23700.

ZHUSSUPOVA, R.; KALIZHANOVA, A. Timeline as a means of digital representation of theoretical material. **Bulletin of L.N. Gumilyov Eurasian National University. Philology Series**, Astana, v. 136, n. 3, 2021. Disponível em: <https://rep.enu.kz/handle/enu/5082>. Acesso em: 23 jul.2023.

GLOSSÁRIO

collider colisor, recurso do Unity.

display apresentação de informação de maneira visual.

drag-and-drop a ação de clicar em um objeto virtual e "arrastá-lo" a uma posição diferente, também conhecida como "arrastar e soltar".

frame uma imagem individual em uma sequência de imagens.

game engine motor de jogo, um software cujo propósito principal é a criação de jogos.

grid grade, recurso do Unity.

placeholder lugar reservado para um propósito específico.

prefab prefabricado, recurso do Unity.

scene cena, recurso do Unity.

script conjunto de instruções para execução de uma função.

slider elemento visual deslizante que controla algo em uma aplicação.

template um modelo predefinido que demonstra a estrutura a ser seguida.

timeline linha do tempo: uma lista de eventos exibidos em ordem cronológica.

APÊNDICE A – SHORT GAME DESIGN DOCUMENT: SCREENER TIMELINE

- **Propósito:** O SCREENER é um jogo educacional para o ensino do processo de Descoberta e Desenvolvimento de Fármacos e Medicamentos (DDF). Enquanto ele é normalmente um jogo de tabuleiro para até seis jogadores, o objetivo deste projeto é criar uma versão condensada dele para um único jogador, visando promover o mesmo aprendizado lúdico e interativo de maneira compacta.
- **Jogo:** O jogo é composto por diversas cartas, cada uma representando uma das etapas do processo de DDF. Uma rodada começa com uma carta qualquer no centro da mesa, e outra carta sendo dada ao jogador. O jogador então coloca sua carta na mesa, na posição que acredita ser adequada à etapa (após, anterior a ou entre as cartas já presentes). A posição escolhida é avaliada e, se o máximo de 3 erros ainda não foi atingido, é então dada uma nova carta, e repete-se o processo até que todas as cartas estejam na mesa, ou um terceiro erro seja cometido. Assim se forma uma linha do tempo do processo de fármacos.

Quadro 1 – SGDD: Screener Timeline

Arte	Programação
Ícone do jogo	Transição entre telas
Background de jogo	Definição do scriptable object das cartas
Botões de jogo	Display das cartas
Menu principal: Logotipo do jogo	Movimentação das cartas
Menu principal: Botões de iniciar, informação e website	Instanciação das cartas
Tela de informação: Janela de regras	Detecção de Game Over
Tela de informação: Botão de Voltar	Contador de tempo
Tela de jogo: Cartas	Detecção de colisão entre cartas em movimento e a zona mesa
Tela de jogo: Área de cartas já jogadas	Avaliação de cartas jogadas
Tela de jogo: Área de cartas não jogadas	Ajustar a posição de cartas posicionadas erroneamente
Tela de jogo: Janela de Game Over	Mudança de interrogação para número após carta ser posicionada
Tela de jogo: Botões de jogar novamente e voltar à tela inicial	Mudança da cor da carta para verde se jogada na posição correta, vermelho caso contrário
Texto	Contador de acertos e erros

APÊNDICE B – RESPOSTAS DO FORMULÁRIO DE AVALIAÇÃO DO SCREENER TIMELINE

As respostas obtidas para o formulário de avaliação do Screener Timeline, exceto aquelas que contêm informações pessoais e comentários, são exibidas neste apêndice.

Figura 23 – Resposta do formulário de avaliação: Data de nascimento

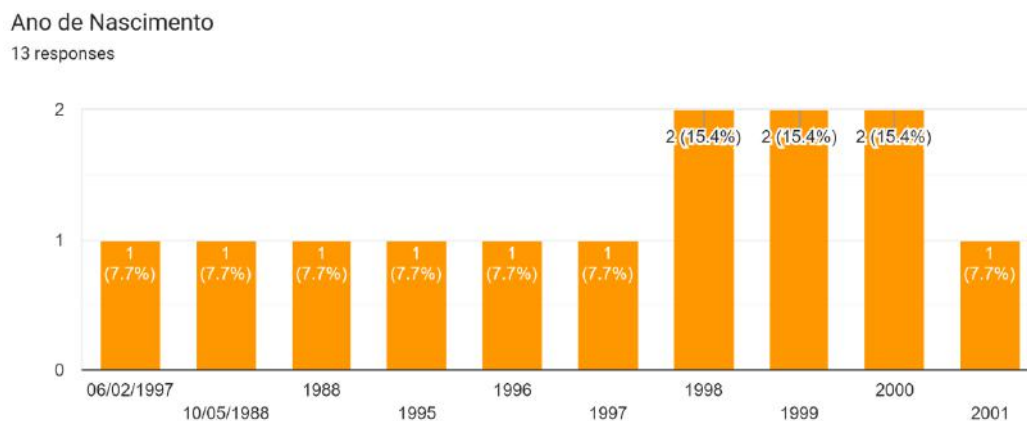


Figura 24 – Resposta do formulário de avaliação: Cidade de residência

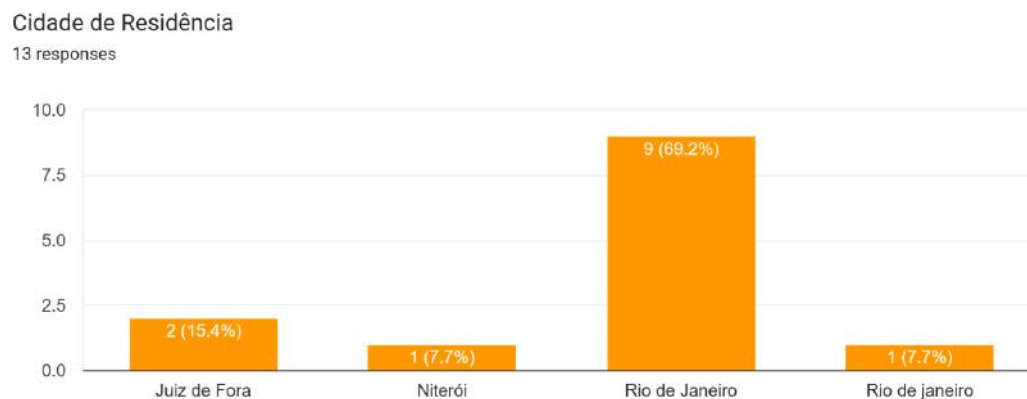


Figura 25 – Resposta do formulário de avaliação: Estado de residência

Estado de Residência

13 responses

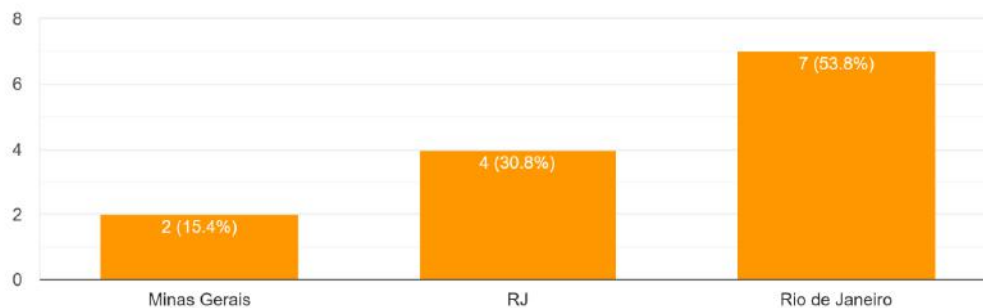


Figura 26 – Resposta do formulário de avaliação: Gênero

Gênero

13 responses

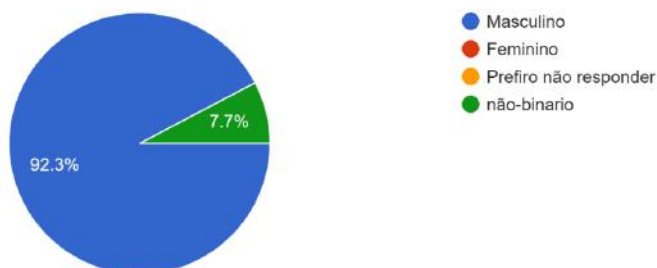


Figura 27 – Resposta do formulário de avaliação: Idade

Faixa Etária

13 responses

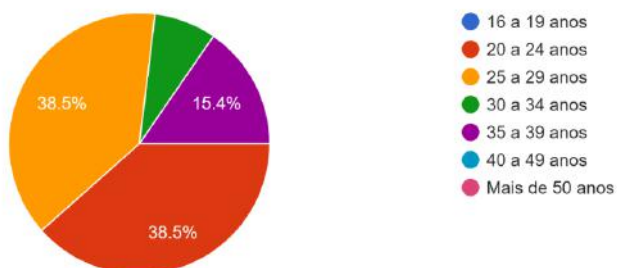


Figura 28 – Resposta do formulário de avaliação: Raça

Com qual raça ou cor você se identifica? ¹ ¹ De acordo com o IBGE

13 responses

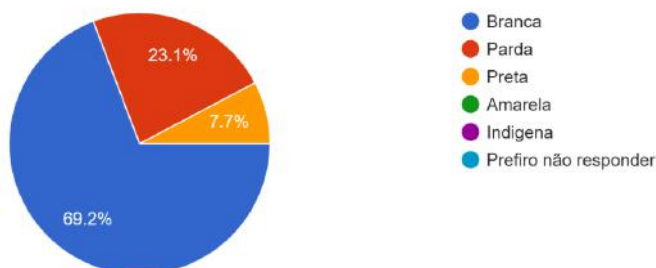


Figura 29 – Resposta do formulário de avaliação: Formação

Qual é a sua área de formação? ² ² De acordo com CNPq/CAPES

13 responses

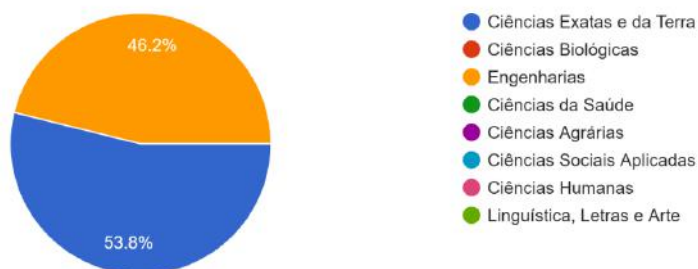


Figura 30 – Resposta do formulário de avaliação: Titulação acadêmica

Qual é a sua maior titulação acadêmica? ¹ ¹ De acordo com o IBGE

13 responses

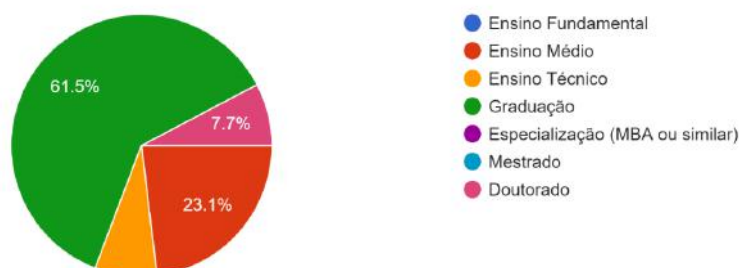


Figura 31 – Resposta do formulário de avaliação: Renda

Qual é a renda mensal total de todos os residentes na sua casa? ¹ ¹ De acordo com o IBGE
13 responses

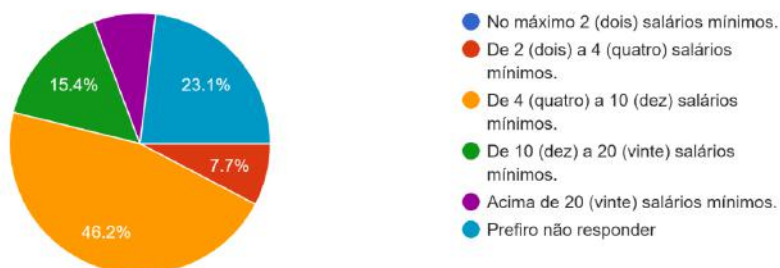


Figura 32 – Resposta do formulário de avaliação: Conhecimento prévio de jogos educacionais

Você sabe o que é um jogo educacional?
13 responses

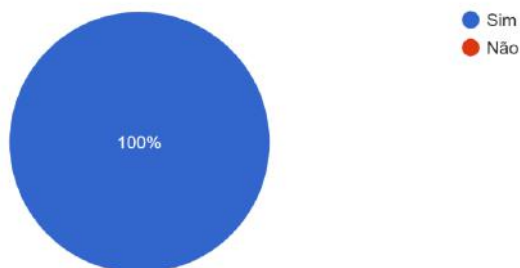


Figura 33 – Resposta do formulário de avaliação: Interação prévia com jogos educacionais

Você já jogou um jogo educacional?
13 responses

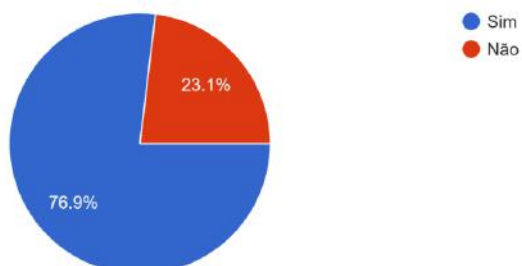


Figura 34 – Resposta do formulário de avaliação: Diversão com jogos educacionais

Você concorda que jogos educacionais são divertidos?

13 responses

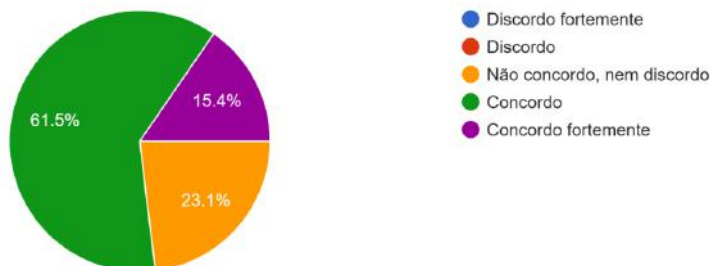


Figura 35 – Resposta do formulário de avaliação: Plataformas de jogo

Em quais plataformas você costuma jogar?

13 responses

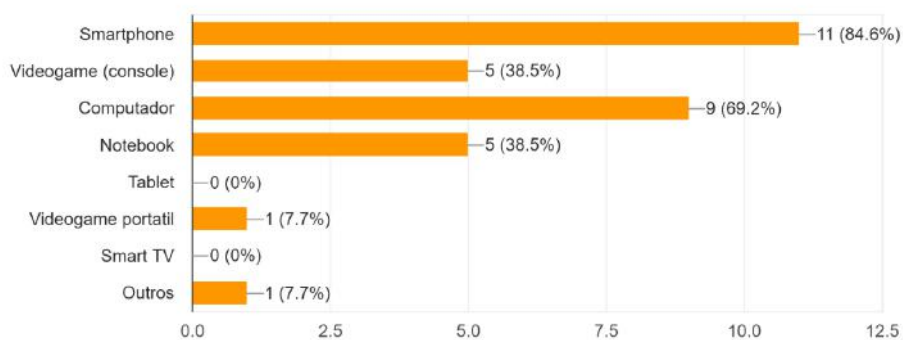


Figura 36 – Resposta do formulário de avaliação: Frequência de jogo

Com qual frequência você costuma jogar jogos digitais (em qualquer uma das plataformas da pergunta acima)?

13 responses

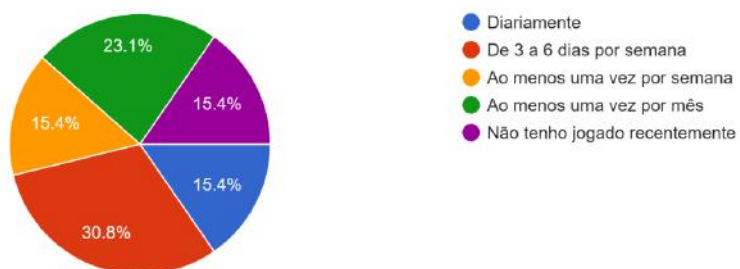
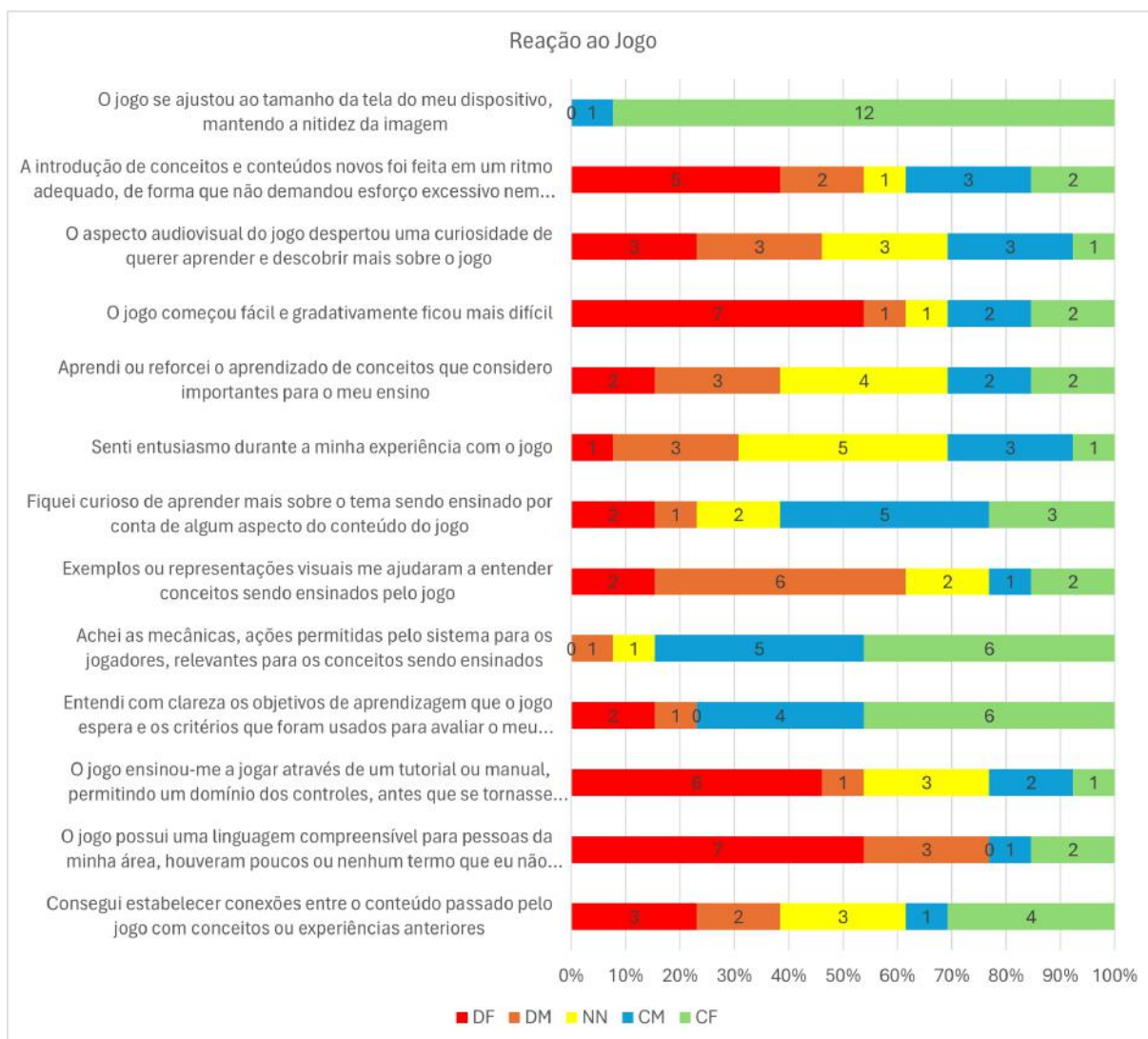


Figura 37 – Resposta do formulário de avaliação: Avaliação da Reação



**ANEXO A – PERGUNTAS DO FORMULÁRIO DE AVALIAÇÃO DO
SCREENER TIMELINE**

1. Inclua o horário de início do preenchimento deste formulário, por favor (Horário)
2. Nome (Texto, opcional)
3. E-mail (Texto, opcional)
4. Ano de Nascimento (Texto)
5. Cidade de Residência (Texto)
6. Estado de Residência (Texto)
7. Gênero
 - a) Masculino
 - b) Feminino
 - c) Prefiro não dizer
 - d) Não-binário
8. Faixa Etária
 - a) 16 a 19 anos
 - b) 20 a 24 anos
 - c) 25 a 29 anos
 - d) 30 a 34 anos
 - e) 35 a 39 anos
 - f) 40 a 49 anos
 - g) Mais de 50 anos
9. Com qual raça ou cor você se identifica?
 - a) Branca
 - b) Parda
 - c) Preta
 - d) Amarela
 - e) Indígena

- f) Prefiro não responder
10. Qual é a sua área de formação?
- a) Ciências Exatas e da Terra
 - b) Ciências Biológicas
 - c) Engenharias
 - d) Ciências da Saúde
 - e) Ciências Agrárias
 - f) Ciências Sociais Aplicadas
 - g) Ciências Humanas
 - h) Linguística, Letras e Arte
11. Qual é a sua maior titulação acadêmica?
- a) Ensino Fundamental
 - b) Ensino Médio
 - c) Ensino Técnico
 - d) Graduação
 - e) Especialização (MBA ou similar)
 - f) Mestrado
 - g) Doutorado
12. Qual é a renda mensal total de todos os residentes na sua casa?
- a) No máximo 2 (dois) salários mínimos
 - b) De 2 (dois) a 4 (quatro) salários mínimos
 - c) De 4 (quatro) a 10 (dez) salários mínimos
 - d) De 10 (dez) a 20 (vinte) salários mínimos
 - e) Acima de 20 (vinte) salários mínimos
 - f) Prefiro não responder
13. Você sabe o que é um jogo educacional?
- a) Sim
 - b) Não
14. Você já jogou um jogo educacional?

- a) Sim
 - b) Não
15. Você concorda que jogos educacionais são divertidos?
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente
16. Em quais plataformas você costuma jogar? (Escolha múltipla)
- a) Smartphone
 - b) Videogame (console)
 - c) Computador
 - d) Notebook
 - e) Tablet
 - f) Videogame portátil
 - g) Smart TV
 - h) Outros
17. Com qual frequência você costuma jogar jogos digitais (em qualquer uma das plataformas da pergunta acima)?
- a) Diariamente
 - b) De 3 a 6 dias por semana
 - c) Ao menos uma vez por semana
 - d) Ao menos uma vez por mês
 - e) Não tenho jogado recentemente
18. Senti entusiasmo durante a minha experiência com o jogo
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente

19. Achei as mecânicas, ações permitidas pelo sistema para os jogadores, relevantes para os conceitos sendo ensinados
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente
20. Consegui estabelecer conexões entre o conteúdo passado pelo jogo com conceitos ou experiências anteriores
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente
21. A introdução de conceitos e conteúdos novos foi feita em um ritmo adequado, de forma que não demandou esforço excessivo nem foi entediante
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente
22. Entendi com clareza os objetivos de aprendizagem que o jogo espera e os critérios que foram usados para avaliar o meu desempenho
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente
23. Fiquei curioso de aprender mais sobre o tema sendo ensinado por conta de algum aspecto do conteúdo do jogo

- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente
24. Exemplos ou representações visuais me ajudaram a entender conceitos sendo ensinados pelo jogo
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente
25. O jogo começou fácil e gradativamente ficou mais difícil
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente
26. O aspecto audiovisual do jogo despertou uma curiosidade de querer aprender e descobrir mais sobre o jogo
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente
27. O jogo possui uma linguagem compreensível para pessoas da minha área, houveram poucos ou nenhum termo que eu não entendi
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo

- d) Concordo
 - e) Concordo fortemente
28. Aprendi ou reforcei o aprendizado de conceitos que considero importantes para o meu ensino
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente
29. O jogo ensinou-me a jogar através de um tutorial ou manual, permitindo um domínio dos controles, antes que se tornasse desafiador
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente
30. O jogo se ajustou ao tamanho da tela do meu dispositivo, mantendo a nitidez da imagem
- a) Discordo fortemente
 - b) Discordo
 - c) Não concordo, nem discordo
 - d) Concordo
 - e) Concordo fortemente