



INSTITUTO FEDERAL

São Paulo

Câmpus Cubatão

**Segurança Confiscada
Projeto de Sistemas**

ESTER FREITAS SILVA
FERNANDA NUNES PIRES DE SOUZA
GABRIEL REZENDE DE OLIVEIRA
JULIA SONALI DOS SANTOS
NICOLLY LIBÂNIO LIMA
RAFAELLA RIOS WALZ
WASHINGTON JOSÉ DA SILVA JUNIOR

SEGURANÇA CONFISCADA

Trabalho de conclusão de curso solicitado pelo docente à disciplina de Projeto de Sistemas lecionada no Instituto Federal de Educação, Ciências e Tecnologia de São Paulo – Câmpus Cubatão. Orientador(a): Prof. Maurício Neves Asenjo

SUMÁRIO

Planejamento	3
1. Pesquisa Inicial e Objetivos	3
2. Roteiro via Google Docs	4
3. Protótipos no Canva	6
Programa	11
4. Funcionalidade	11
5. Como baixar	13
Sistema	16
6. Funcionamento do jogos	16
7. Acessibilidade aplicada	20
Legenda	22
8. Como foi feito	22
9. Como foi aplicado	22
Controles	24
10. Como foi feito	24
11. Como foi aplicado	26
Desenhos	29
12. Como foi feito (Krita e Capcut)	29
13. Como foi aplicado (para rodar os vídeos exemplo)	29
Dublagens	31
14. Como foi feito	31
15. Como foi aplicado	31
Fase 1	33
16. Projeto de fase	33
17. Aplicações	34
18. Como jogar	37
Fase 2	38
19. Projeto de fase	38
20. Aplicações	39
21. Como jogar	42
Fase 3	43
22. Projeto de fase	43
23. Aplicações	44
24. Como jogar	47

Planejamento

1. Pesquisa Inicial e Objetivos

A pesquisa que fundamenta este projeto aborda a viabilidade e os desafios de criar um jogo inclusivo para diferentes tipos de público. O foco principal é explorar as questões que envolvem o processo de criação de um jogo inclusivo, os principais obstáculos enfrentados durante o desenvolvimento e as soluções possíveis para promover a inclusão no design e na jogabilidade.

Para viabilizar esta análise, foi necessário estabelecer metas claras que orientassem o desenvolvimento do projeto. Assim, definimos um objetivo geral e objetivos específicos, descritos a seguir:

Objetivo Geral:

Contribuir para possibilidade de inclusão no desenvolvimento de jogos eletrônicos.

Objetivos Específicos:

Compreender as demandas de acessibilidade do público-alvo dos jogos eletrônicos;

Identificar dificuldades de aplicação da acessibilidade nos jogos eletrônicos;

Esquematizar demandas para aplicar ferramentas de acessibilidade em um protótipo.

2. Roteiro via Google Docs

Para as legendas e a história do jogo, utilizamos o Google Docs, uma ferramenta de edição e visualização de documentos online. Essa escolha permitiu uma organização eficiente, já que os arquivos podiam ser acessados e editados de qualquer dispositivo conectado à internet.

Dividimos as informações em diferentes documentos para facilitar a identificação e o acesso. Os principais arquivos utilizados foram:

1. História do Jogo: Continha toda a narrativa do jogo, incluindo algumas imagens representativas das fases e personagens.
2. Dublagem: Este arquivo foi dedicado aos trechos que deveriam ser gravados, com instruções detalhadas sobre a forma de interpretação de cada trecho.

Como Acessar o Google Docs pela Primeira Vez

Pelo Computador:

1. Abra o Navegador
 - No navegador de sua preferência, vá para <https://docs.google.com>.
2. Faça Login ou Crie uma Conta do Google
 - Caso já tenha uma conta Google (Gmail), insira seu endereço de e-mail e senha.
 - Se não tiver uma conta, clique em “Criar conta” e preencha os dados solicitados:
 - Nome completo;
 - Nome de usuário (o endereço de e-mail);
 - Senha e confirmação de senha.
 - Siga os passos para configurar sua nova conta.
3. Conceda Permissões
 - No primeiro acesso, o Google Docs pode solicitar permissões de uso (como acesso aos seus arquivos no Google Drive). Aceite para prosseguir.
4. Crie seu Primeiro Documento
 - Clique no botão “+” (Documento em Branco) para começar a criar seu primeiro arquivo ou explore os templates disponíveis.

Pelo Celular:

1. Baixe o Aplicativo Google Docs
 - Abra a Google Play Store (Android) ou Apple Store (iOS), procure por “Google Docs” e instale o aplicativo.
2. Abra o Aplicativo e Faça Login
 - Ao abrir o app, insira os dados da sua conta Google.
 - Caso não tenha uma conta, toque em “Criar conta” e siga as instruções:

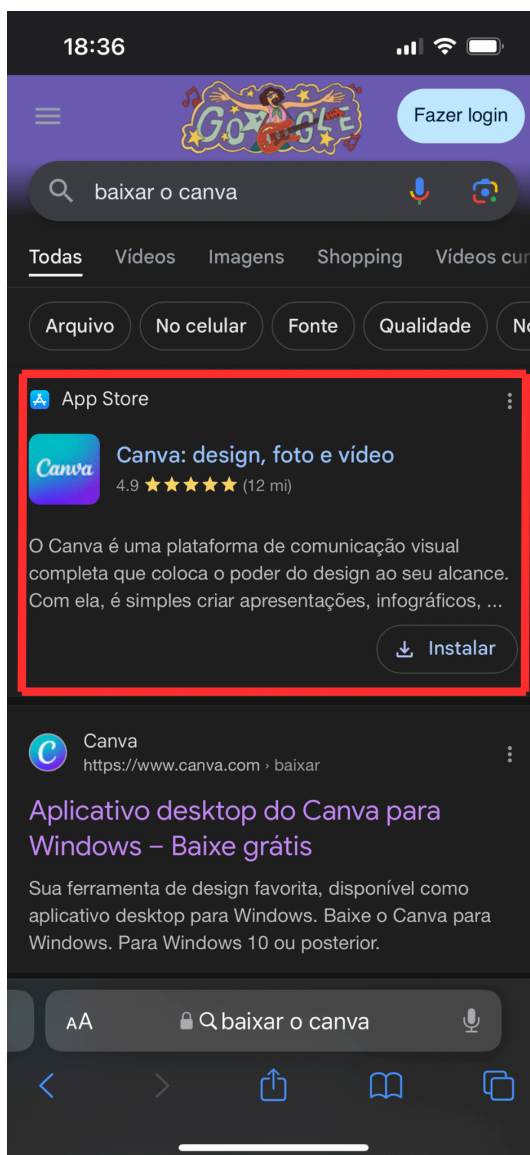
- Adicione seu nome completo, crie um e-mail e uma senha.
 - Conclua o processo de verificação (se solicitado).
3. Permita o Acesso aos Arquivos
- O aplicativo pode solicitar permissões para acessar arquivos no dispositivo e no Google Drive. Permita para prosseguir.
4. Crie Seu Primeiro Documento
- Toque no botão “+” (Documento em Branco) no canto inferior direito e comece a editar.

3. Protótipos no Canva

Para o design do projeto, utilizamos o aplicativo gratuito de design gráfico Canva, que nos permitiu criar esboços das fases do jogo com maior liberdade. Essa ferramenta facilitou tanto o processo de design quanto a orientação sobre o funcionamento das fases do jogo.

Para baixar o Canva, siga os passos abaixo:

1. Acesse o Google e pesquise “Baixar o Canva”.
- Para dispositivos Android, clique na opção que direciona ao Google Play.
- Para dispositivos iOS, selecione a opção da Apple Store.



2. Alternativamente, acesse diretamente o site oficial do Canva pelo link: https://www.canva.com/pt_br/baixar.

- Role a página até encontrar a opção “Baixar”, escolha o sistema operacional desejado e siga as instruções de instalação.

18:36



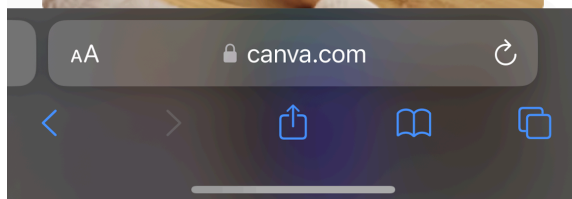
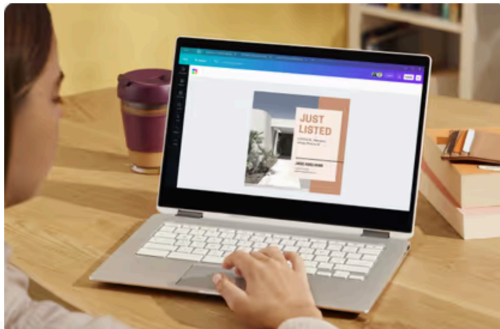
Canva para Windows

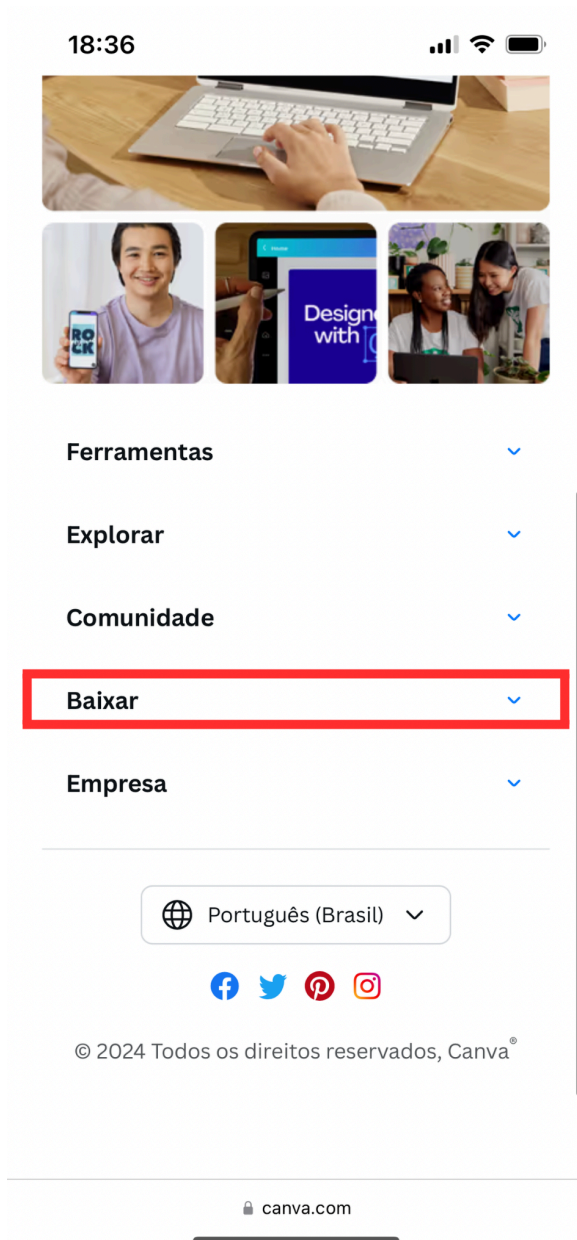
Sua ferramenta de design favorita,
disponível como aplicativo desktop para
Windows.

Baixe o Canva para Windows

Para Windows 10 ou posterior.

Disponível também em [Mac OS](#), [iOS](#) e [Android](#).





Crie uma conta no aplicativo, através dos seguintes passos:

2. Clique em “Registrar”
- No site ou no app, localize o botão “Registrar” ou “Começar Agora” na página inicial.
3. Escolha a Forma de Registro

- O Canva oferece diferentes maneiras de criar uma conta:
 - Google: Use sua conta do Gmail.
 - Facebook: Conecte-se com sua conta do Facebook.
 - E-mail: Insira manualmente seu endereço de e-mail e crie uma senha.
4. Preencha os Dados (se necessário)
- Caso opte por criar a conta com e-mail, insira:
 - Nome completo;
 - Endereço de e-mail válido;
 - Uma senha segura.
5. Confirme o Cadastro
- Após o registro, você pode receber um e-mail de confirmação. Acesse sua caixa de entrada, localize o e-mail do Canva e clique no link para ativar sua conta.
6. Personalize sua Conta (Opcional)
- Ao entrar pela primeira vez, o Canva pode perguntar sobre seu objetivo principal (trabalho, estudo, uso pessoal). Escolha a opção que mais se adequa ao seu caso.
7. Comece a Usar
- Após criar a conta, você será redirecionado para a página inicial do Canva, onde poderá começar a criar designs imediatamente.

Programa

4. Funcionalidade

O Visual Studio é um ambiente de desenvolvimento integrado (IDE) poderoso e amplamente utilizado para a criação de aplicativos de software. Ele oferece uma ampla gama de funcionalidades para suportar desenvolvedores em diversas fases do desenvolvimento de software, incluindo codificação, depuração, testes e implantação. Abaixo estão algumas das principais funcionalidades do Visual Studio:

1. Edição de Código:

Realce de Sintaxe: Visual Studio oferece realce de sintaxe para várias linguagens de programação como C#, C++, JavaScript, Python, HTML, CSS, entre outras, facilitando a leitura e a escrita de código.

IntelliSense: Recurso de autocompletar e sugestões inteligentes que ajudam a acelerar o processo de codificação, sugerindo métodos, classes e variáveis enquanto o desenvolvedor digita.

Refatoração: Ferramentas que ajudam a melhorar e otimizar o código, como renomeação de variáveis, extração de métodos e reorganização de código, sem alterar seu comportamento.

2. Depuração:

Depurador Visual: Permite ao desenvolvedor encontrar e corrigir erros (bugs) no código. O depurador pode ser utilizado para definir pontos de interrupção (breakpoints), examinar o valor de variáveis durante a execução e realizar uma análise detalhada do fluxo do programa.

Análise de Memória: Oferece ferramentas para analisar o uso de memória da aplicação e encontrar vazamentos de memória.

3. Controle de Versão (Git e TFS):

Integração com Git: Permite o uso de Git diretamente dentro do Visual Studio para controle de versão, com suporte a repositórios locais e remotos (GitHub, Azure Repos, Bitbucket).

Team Foundation Server (TFS): O Visual Studio também oferece integração com o TFS, permitindo gerenciar tarefas, bugs e realizar controle de versão em projetos corporativos.

4. Design de Interface (UI):

Designer Gráfico: Permite que os desenvolvedores criem interfaces gráficas para aplicativos desktop (Windows Forms, WPF), web (ASP.NET) ou aplicativos móveis de maneira intuitiva, usando o designer visual.

Windows Forms e WPF: Ferramentas de design para criar interfaces ricas para aplicativos desktop, com suporte a arrastar e soltar controles.

5. Compilação e Implantação:

Compilação Rápida: O Visual Studio oferece compilação incremental para acelerar o processo de construção de aplicativos, garantindo que apenas os arquivos alterados sejam recompilados.

Implantação na Nuvem: Suporte a implantações para plataformas de nuvem como o Azure, com ferramentas para configurar e lançar aplicativos diretamente de dentro da IDE.

6. Testes:

Testes Unitários: Suporta a criação e execução de testes unitários para garantir que os métodos individuais do código funcionem corretamente. O Visual Studio oferece integração com frameworks populares como MSTest, NUnit e xUnit.

Testes de Interface de Usuário: Oferece recursos para automação de testes de UI, permitindo testar interfaces gráficas e interações com o usuário.

7. Extensibilidade:

Extensões e Plugins: O Visual Studio tem um vasto mercado de extensões (Visual Studio Marketplace) que permite aos desenvolvedores adicionar novas funcionalidades à IDE, como suporte para novas linguagens, ferramentas de produtividade e templates de projeto.

Ferramentas Personalizáveis: Possibilidade de personalizar o ambiente de desenvolvimento, incluindo atalhos de teclado, temas, barras de ferramentas e layouts.

8. Suporte Multiplataforma:

Desenvolvimento para Windows, Linux e macOS: Embora o Visual Studio seja mais conhecido pelo desenvolvimento para o ecossistema Windows, ele também oferece suporte ao desenvolvimento para outras plataformas, especialmente com o uso do .NET Core, Xamarin e ferramentas de Docker.

Desenvolvimento Mobile (Xamarin): O Visual Studio tem integração com o Xamarin, permitindo que os desenvolvedores criem aplicativos móveis nativos para iOS e Android usando C#.

9. Ferramentas de Análise de Código:

Analisador de Código: O Visual Studio pode detectar problemas no código com base em regras predefinidas ou personalizadas, ajudando os desenvolvedores a manter a qualidade do código e aderir a boas práticas.

Análise de Desempenho: Oferece ferramentas para monitorar e otimizar o desempenho do aplicativo, como o Profiler, que pode identificar gargalos de desempenho em tempo de execução.

10. Suporte à Programação em Equipe:

Azure DevOps: Integração com o Azure DevOps para coordenar projetos, acompanhar problemas e bugs, e gerenciar o ciclo de vida do desenvolvimento em equipes.

Ferramentas de Colaboração: Suporte a comunicação e colaboração em tempo real entre os membros da equipe, com recursos como Live Share, onde os desenvolvedores podem editar o código juntos em tempo real.

5. Como baixar

VISUAL STUDIO

1. Escolha da edição do Visual Studio

O Visual Studio oferece diferentes edições:

- Community: Gratuita e ideal para estudantes, iniciantes e projetos pessoais.
- Professional: Pago, indicado para desenvolvedores profissionais.
- Enterprise: Pago, voltado para grandes empresas.


Recomendação: Use o Visual Studio Community se estiver começando.

2. Baixar o instalador

- Acesse o site oficial: <https://visualstudio.microsoft.com/>.
- Clique em Download na edição desejada (geralmente “Community”).
- O instalador será baixado no seu computador como um arquivo executável (ex.: VisualStudioSetup.exe).

3. Execute o instalador


- Localize o arquivo baixado no seu computador.
- Clique duas vezes no arquivo VisualStudioSetup.exe para iniciar.
- Na tela inicial, você verá uma opção para continuar com a instalação.


Visual Studio
<https://visualstudio.microsoft.com> > pt-br > downloads


Downloads - Visual Studio - Microsoft

3 de set. de 2024 — Baixe o VS Code ou IDE do **Visual Studio** gratuitamente. Experimente as edições do **Visual Studio** Professional ou Enterprise no Windows e Mac.

[Visão geral](#) · [Comparar Edições](#) · [Downloads mais antigos](#)



Downloads



Visual Studio 2022 | Windows

O IDE mais abrangente para desenvolvedores .NET e C++ no Windows para criação de web, nuvem, desktop, aplicativos móveis, serviços e jogos.

Versão prévia

Obtenha acesso antecipado aos recursos mais recentes que ainda não estão na versão principal

[Saiba mais →](#)

Comunidade

IDE poderoso, gratuito para estudantes, colaboradores de código aberto e indivíduos

Download gratuito

Profissional

IDE profissional mais adequado para equipes pequenas

[Teste gratuito](#)

Enterprise

Solução escalável e completa para equipes de qualquer tamanho

[Teste gratuito](#)

[Notas sobre a versão →](#)
[Comparar Edições →](#)
[Como fazer a instalação offline →](#)
[Termos de Licença →](#)

[Comentários ↗](#)

Salvar como
✕

← → ⌵ ⬆
Área de Trabalho
↻
Pesquisar em Área de Traba... 🔍

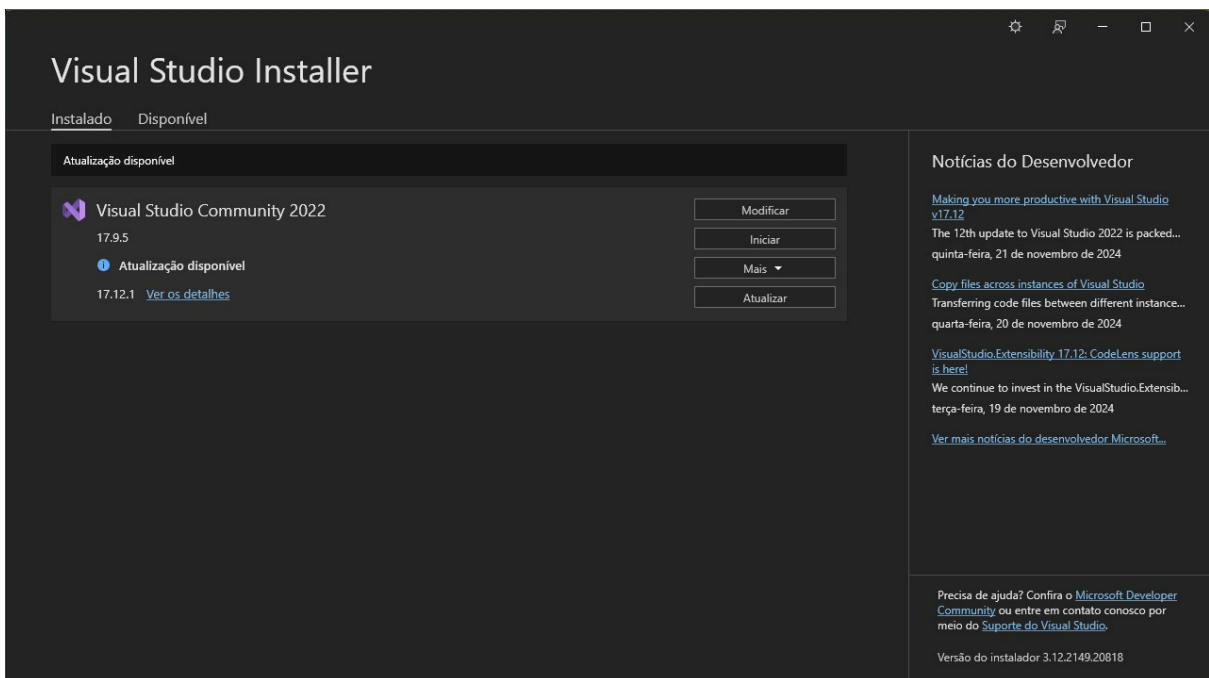
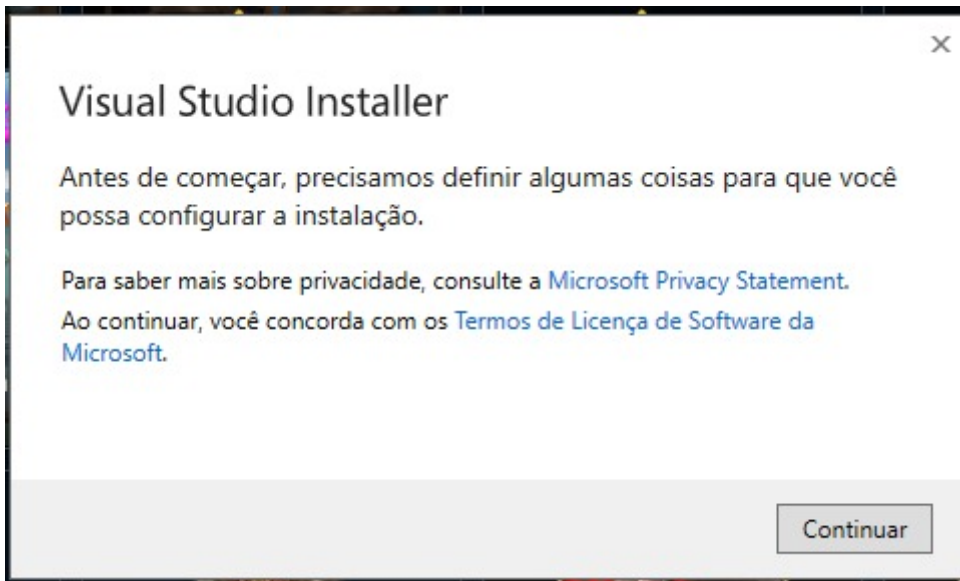
Organizar Nova pasta

	Nome	Data de modificação	Tipo	Taman
	Início			
	Galeria			
	Área de Trab			
	Downloads			
	Documentos			
	Aplicativoss	05/11/2024 22:24	Pasta de arquivos	
	atividades	17/10/2024 19:27	Pasta de arquivos	
	Joguinhoss	17/10/2024 19:27	Pasta de arquivos	
	Mouse-Fade-Software-v2	24/10/2024 18:41	Pasta de arquivos	
	Nova pasta	01/11/2024 18:45	Pasta de arquivos	

Nome:

Tipo:

⬆ Ocultar pastas
Salvar
Cancelar

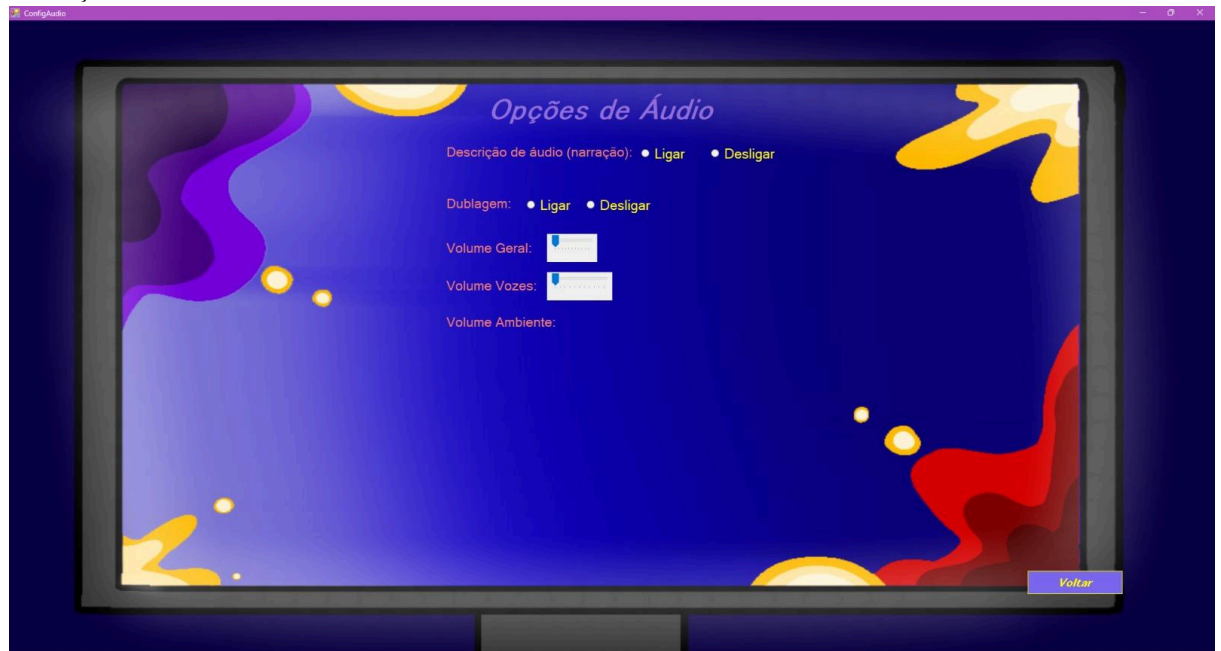


Sistema

6. Funcionamento do jogos

Configurações de Áudio e Vídeo:

Controles dedicados permitem ajustar parâmetros de áudio, como ativação/desativação de dublagem e narração.



A configuração de dublagem e descrição de áudio foi implementada utilizando if e else, como mostra abaixo:


```

private void Rdb_LigarDublagem_CheckedChanged(object sender, EventArgs e)
{
    if (Rdb_LigarDublagem.Checked)
    {
        Config.Dub = true;
    }
}

private void radioButton1_CheckedChanged(object sender, EventArgs e)
{
    if (Rdb_ligarNar.Checked)
    {
        Config.DescAudi = true;
    }
}

private void Rdb_DesligarNarração_CheckedChanged(object sender, EventArgs e)
{
    if (Rdb_DesligarNarração.Checked)
    {
        Config.DescAudi = false;
    }
}

private void Rdb_DesligarDublagem_CheckedChanged(object sender, EventArgs e)
{
    if (Rdb_DesligarDublagem.Checked)
    {
        Config.Dub = false;
    }
}

```

Manipulação de Imagens com PictureBoxes:

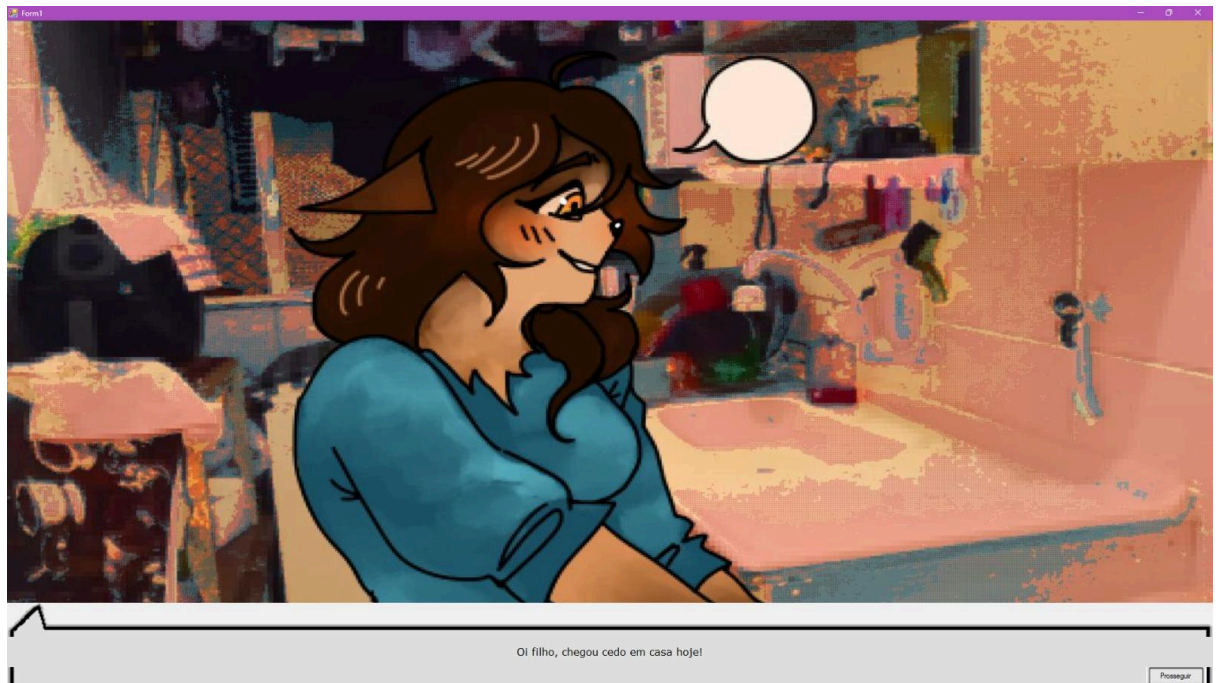
Utiliza diversas PictureBoxes para alterar imagens dinamicamente durante a execução do programa, possibilitando uma interface visual rica e adaptável.

Reprodução Multimídia com Windows Media Player:

A extensão do Windows Media Player é utilizada para a reprodução de cutscenes, oferecendo suporte a vídeos integrados diretamente na aplicação.

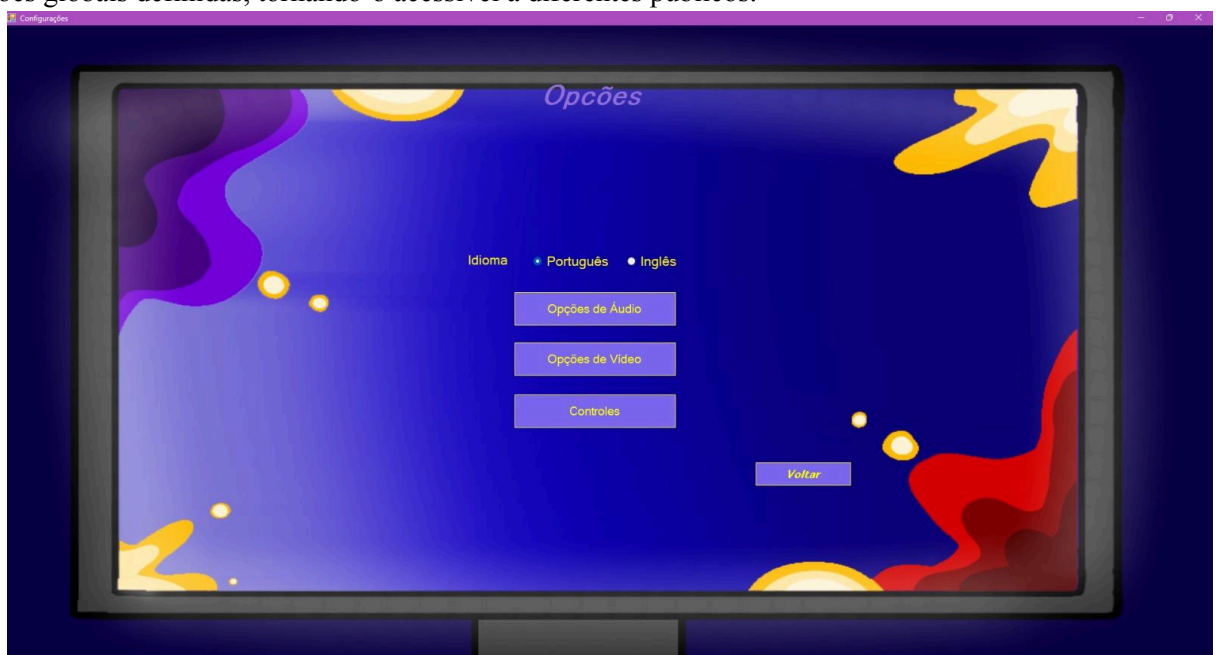
Gerenciamento de Legendas:

O sistema implementa legendas utilizando TextBoxes, permitindo que o texto seja exibido de forma sincronizada com as cutscenes, garantindo acessibilidade e clareza ao conteúdo.



Internacionalização:

O projeto suporta dois idiomas, com ajustes automáticos de interface dependendo das configurações globais definidas, tornando-o acessível a diferentes públicos.



```

private void ConfigAudio_Load(object sender, EventArgs e)
{
    if (Config.Ling == true)
    {
        lbl_titulo.Text = "Opções de Áudio";
        lbl_narracao.Text = "Descrição de áudio (narração):";
        lbl_dublagem.Text = "Dublagem:";
        lbl_geral.Text = "Volume Geral:";
        lbl_vozes.Text = "Volume Vozes:";
        lbl_ambiente.Text = "Volume Ambiente:";
        Rdb_LigarDublagem.Text = "Ligar";
        Rdb_DesligarDublagem.Text = "Desligar";
        Rdb_DesligarNarração.Text = "Desligar";
        Rdb_ligarNar.Text = "Ligar";
    }
    else
    {
        lbl_titulo.Text = "Audio Settings";
        lbl_narracao.Text = "Audio Description (narration):";
        lbl_dublagem.Text = "Dubbing:";
        lbl_geral.Text = "Overall Volume:";
        lbl_vozes.Text = "Voices Volume:";
        lbl_ambiente.Text = "Ambient Volume:";
        Rdb_LigarDublagem.Text = "On";
        Rdb_DesligarDublagem.Text = "Off";
        Rdb_DesligarNarração.Text = "Off";
        Rdb_ligarNar.Text = "On";
    }
}

```

Navegação e Interface Amigável:

A aplicação possui transições fluidas entre formulários, como menus de configurações e janelas multimídia, sempre mantendo o foco em uma experiência de usuário intuitiva e centralizada.

7. Acessibilidade aplicada

A acessibilidade no jogo foi aplicada com base em práticas que garantem que os jogadores tenham uma experiência inclusiva, independentemente de possíveis limitações físicas ou sensoriais. Abaixo estão os pontos que evidenciam a aplicação de acessibilidade no jogo:

Legendas para Narrativas e Cutscenes

Uso de TextBox para Legendas:

O jogo implementou legendas para garantir que jogadores com limitações auditivas possam acompanhar os diálogos e eventos das cutscenes. Esses textos aparecem de forma sincronizada com os vídeos e eventos da narrativa, utilizando controles como `TextBox` para exibir as falas.

As legendas são configuradas dinamicamente com base no conteúdo do vídeo, proporcionando uma experiência fluida e acessível.

Elementos Gráficos e Interfaces Visuais

Labels Utilizados como Títulos e Instruções

Labels são usados como títulos e instruções claras, ajudando os jogadores a entenderem o propósito de cada tela ou fase. Isso beneficia especialmente jogadores com dificuldades de leitura rápida ou que preferem informações visuais diretas.

PictureBoxes para Contexto Visual:

O uso de imagens nos `PictureBox`, como fundos e elementos visuais, contribui para o contexto do jogo, garantindo que os jogadores consigam interpretar as situações com apoio visual, caso tenham dificuldades em processar textos longos.

Menus e Botões de Navegação Intuitivos

- Botões para Controle e Navegação:

O jogo inclui botões bem posicionados para ações específicas, como "Iniciar Jogo", "Opções", e seleção de decisões durante as fases. Isso facilita a interação para jogadores que utilizam dispositivos como teclado ou mouse.

Navegação Consistente:

A utilização de botões para passar cutscenes e interagir durante as fases mantém uma navegação consistente, reduzindo a confusão e proporcionando acessibilidade cognitiva.

Suporte à Compreensão de Escolhas

Durante as fases, o jogo apresenta escolhas claras e objetivas, como botões que indicam "Pedir Ajuda", "Ignorar" ou "Bloquear", representando ações que o jogador pode tomar. Essa estrutura com texto direto ajuda jogadores com dificuldades cognitivas ou dislexia, pois evita ambiguidades.

Configuração de Áudio e Vídeo

Configuração para Vídeos Sem Dublagem:

Caso a dublagem (Config.Dub) esteja desativada, o jogo garante que as legendas fiquem habilitadas para fornecer contexto sobre a narrativa.

Controle de Erros e Feedback ao Usuário

Mensagens de Erro Claras:

Quando ocorrem falhas no carregamento de vídeos ou outros elementos, o jogo exibe mensagens de erro compreensíveis, evitando que o jogador fique perdido. Isso é especialmente útil para pessoas com dificuldades técnicas ou que dependem de feedback imediato.

Configuração de Temporização e Fluxo:

A temporização das cutscenes, legendas e transições foi ajustada para que o jogador tenha tempo suficiente de absorver o conteúdo, garantindo acessibilidade a pessoas que processam informações em ritmos diferentes.

O jogo foi planejado para atender uma ampla gama de necessidades dos jogadores. Desde o suporte a limitações auditivas com legendas, passando por uma interface visual clara e menus acessíveis, até feedbacks informativos para evitar confusões, o design do jogo reflete a preocupação com inclusão. O uso de botões intuitivos, imagens de apoio, legendas bem sincronizadas e mensagens claras faz com que o jogo seja acessível a uma audiência diversificada.

Legenda

8. Como foi feito

Para criar, foi necessário consultar primeiro o documento contendo a história do jogo juntamente com as falas de cada personagem, para assim conseguirmos implementar cada legenda em seu devido tempo.

9. Como foi aplicado

Utilizando as labels e textbox do Visual Studio como principais ferramentas, aplicamos as legendas nas cutscenes com a implementação de um timer, responsável pela troca de texto conforme o vídeo é rodado. Nas demais situações foi utilizado apenas a tecla ENTER do teclado para troca de falas e

conversações.

```
private void Timer_Tick(object sender, EventArgs e)
{
    try
    {
        // Verifica se o controle está acessível
        if (axWindowsMediaPlayer2 != null && axWindowsMediaPlayer2.Ctlcontrols != null)
        {
            currentSecond = (int)axWindowsMediaPlayer2.Ctlcontrols.currentPosition;

            if (Config.Leg)
            {
                // Verifica o tempo decorrido e atualiza a label
                if (currentSecond >= 0 && currentSecond < 9)
                {
                    if (Config.Ling == true)
                    {
                        lbl_legenda.Text = "Pitico escuta uma risada vindo do PC e um portal se abre a sua frente, surgindo um anzol de pesca o agarrando pela camisa e o jogando para dentro";
                    }
                    else
                    {
                        lbl_legenda.Text = "Pitico hears a laugh coming from the PC and a portal opens in front of him, with a fishing hook appearing, grabbing him by the shirt and throwing him inside.";
                    }
                }
                else if (currentSecond >= 9 && currentSecond < 14)
                {
                    if (Config.Ling == true)
                    {
                        lbl_legenda.Text = "HAHAHAHAHAHA Você realmente caiu nessa? Como você é ingênuo garoto...";
                    }
                    else
                    {
                        lbl_legenda.Text = "HAHAHAHAHAHA Did you really fall for that? How naive you are boy...";
                    }
                }
                else
                {
                    lbl_legenda.Text = ""; // Limpa a legenda após 14 segundos
                    timer.Stop(); // Para o temporizador
                }
            }
        }
    }
    catch (InvalidComObjectException ex)
    {
        // Lidar com o erro, por exemplo, registrar ou mostrar uma mensagem
        MessageBox.Show("Erro ao acessar o reprodutor de mídia: " + ex.Message);
        timer.Stop(); // Para o temporizador se ocorrer um erro
    }
}

private void PositionLegenda()
{
    lbl_legenda.Left = (this.ClientSize.Width - lbl_legenda.Width) / 2;
    lbl_legenda.Top = this.ClientSize.Height - lbl_legenda.Height - 10;

    this.Resize += (s, e) =>
    {

```

Controles

10. Como foi feito

Controles do Jogo

Os controles foram desenvolvidos utilizando componentes básicos e recursos do Windows Forms (WinForms), proporcionando uma experiência interativa e funcional para os jogadores. A seguir, está a explicação de como cada controle foi aplicado:

Botões (Button)

Onde foram utilizados:

Nos menus principais, para ações como iniciar o jogo, acessar opções ou sair.

Durante as cutscenes, para avançar o vídeo ou tomar decisões na narrativa.

Nas fases, como opções interativas para o jogador escolher ações específicas, como na fase 2.

Como foram implementados:

Cada botão foi configurado com um evento Click, que chama funções específicas, como navegação entre formulários, execução de ações durante as fases ou controle da reprodução de vídeos.

Exemplo: Um botão para avançar cutscenes chama uma função que muda o estado atual da narrativa e carrega o próximo conteúdo.

Label

Onde foram utilizados:

Como títulos em telas principais, indicando a fase ou a seção atual.

Para instruções claras nas telas de interação, como menus e fases.

Como foram implementados:

Labels foram estilizados com fontes personalizadas, tamanhos e cores adequados para serem facilmente legíveis.

Exemplo: O título da fase é exibido no topo do formulário, indicando o progresso do jogador.

TextBox

Onde foram utilizados:

Para exibir legendas sincronizadas durante as cutscenes, permitindo que o jogador acompanhe o diálogo ou a narrativa.

Como foram implementados:

O conteúdo da TextBox é atualizado dinamicamente conforme o vídeo avança, sincronizando as legendas com os eventos da cutscene.

As caixas de texto foram configuradas como somente leitura (ReadOnly = true) para evitar interferências do jogador.

PictureBox

Onde foram utilizados:

Para fundos visuais, compondo a estética das telas e fases.

Para exibir personagens ou cenários, tornando as interações mais visuais e dinâmicas.

Como foram implementados:

Imagens foram carregadas nas PictureBox através de recursos (Resources) ou arquivos externos.

Exemplo: Nas fases, imagens de fundo foram ajustadas para preencher o formulário, criando uma ambientação imersiva.

Windows Media Player (AxWindowsMediaPlayer)

Onde foi utilizado:

Para reproduzir as cutscenes do jogo.

Como foi implementado:

O componente foi configurado para carregar vídeos a partir de recursos do projeto (Resources) ou arquivos temporários.

Exemplo: Durante as cutscenes, o controle é ativado e desativado dinamicamente para gerenciar a exibição dos vídeos.

11. Como foi aplicado

Utilizamos a tecla enter para avançar os formulários dentro do nosso jogo, com o código mostrado abaixo:

```
protected override bool ProcessCmdKey(ref Message msg, Keys keyData)
{
    if (keyData == Keys.Enter)
    {
        Form historiaForm = new História();
        historiaForm.Show();
        this.Hide();
    }
}
```

Também utilizamos botões para entregar mais opções de avanço, seja pra passar uma legenda ou também pra passar propriamente de formulário, como mostra abaixo:

```
private void button1_Click(object sender, EventArgs e)
{
    this.Hide();

    foreach (Form form in Application.OpenForms)
    {
        if (form is FormAnuncio)
        {
            form.BringToFront();
            return;
        }
    }

    Form proximoFormulario = new FormAnuncio();
    proximoFormulario.Show();
}
```

Botões:

Cutscenes: Botões foram aplicados para avançar as cutscenes, permitindo que o jogador controle o ritmo da narrativa de maneira simples e interativa.

```

private void button1_Click(object sender, EventArgs e)
{
    this.Hide();

    foreach (Form form in Application.OpenForms)
    {
        if (form is FormAnuncio)
        {
            form.BringToFront();
            return;
        }
    }

    Form proximoFormulario = new FormAnuncio();
    proximoFormulario.Show();
}

```

Os botões foram essenciais para que o jogador pudesse realizar escolhas durante o gameplay, sendo cada botão associado a uma ação específica. No menu principal e nas opções do jogo, os botões desempenham um papel funcional, como iniciar o jogo, acessar configurações ou navegar entre diferentes telas.

```

private void btn_iniciar_Click(object sender, EventArgs e)
{
    this.Hide();
    Form frm = new Inicio();
    frm.Closed += (s, args) => this.Close(); // Fecha o Menu quando o formulário Inicio for fechado
    frm.Show();
}

```

Labels:

Utilizados como títulos e elementos descritivos em diversas telas do jogo.

Servem para orientar o jogador, destacando informações importantes, como o nome da fase, opções no menu ou instruções de gameplay.

TextBox:

Implementados para exibir as legendas durante as cutscenes e as fases do jogo.

As legendas contribuem para o aspecto narrativo, garantindo que o jogador compreenda os diálogos e os eventos apresentados.

```

private void ExibirLegenda()
{
    switch (legendaIndex)
    {
        case 0:
            if (Config.Ling == true)
            {
                legendaAtual = "Pitico o jovem guaxinim chega em casa cansado depois de um longo dia estudando e vai direto a cozinha conversar com sua mãe, que está preparando o almoço.";
            }
            else
            {
                legendaAtual = "Pitico, the young raccoon comes home tired after a long day of studying and goes straight to the kitchen to talk to his mother, who is preparing lunch.";
            }
            break;

        case 1:
            if (Config.Ling == true)
            {
                legendaAtual = "Após o almoço, Pitico se dirige ao seu quarto, deixando as suas coisas de lado e indo ligar seu computador.";
            }
            else
            {
                legendaAtual = "After lunch, Pitico goes to his room, leaving his things aside and going to turn on his computer.";
            }
            break;

        default:
            break;
    }

    lbl_fala2.Text = legendaAtual;
}

private void LegendaTimer_Tick(object sender, EventArgs e)
{
    if (axWindowsMediaPlayer1 != null && !axWindowsMediaPlayer1.IsDisposed)
    {
        double currentTime = axWindowsMediaPlayer1.Ctlcontrols.currentPosition;

        if (videoAtivo == "video")
        {
            if (currentTime >= 0 && currentTime < 12)
            {
                legendaIndex = 0;
                ExibirLegenda();
            }
        }
        else if (videoAtivo == "cut2")
        {
            if (currentTime >= 0 && currentTime < 12)
            {
                legendaIndex = 1;
                ExibirLegenda();
            }
        }
    }
}
}

```

PictureBox:

Fundos dos Formulários: Usados para adicionar imagens de fundo aos formulários, proporcionando uma ambientação visual ao jogo.

Sprites dos Personagens e Elementos Visuais: PictureBoxes também foram usados para exibir os personagens (como Pitico e o CSCHAR) e outros elementos gráficos, garantindo flexibilidade para manipulação e mudanças dinâmicas durante o jogo.

Botões: Decisões e navegação.

Labels: Informações e orientações.

TextBox: Comunicação narrativa.

PictureBox: Representação visual e imersão.

Desenhos

12. Como foi feito (Krita e Capcut)

As artes foram rascunhadas em papel, e então transferidas para plataformas de desenho digital gratuitas, como o *IBIS Paint mobile*, usando a touchscreen, ou com o *Krita* usando uma mesa digitalizadora *Bamboo Wacom*. Após a confecção das ilustrações, suas imagens eram colocadas no programa de edição também gratuito, o *Capcut*, utilizando as ferramentas de corte, efeitos e transições gratuitas.

13. Como foi aplicado (para rodar os vídeos exemplo)

No código:

O código exibido utiliza vídeos dentro de um jogo ou aplicação C# com o componente *AxWindowsMediaPlayer* para reproduzir arquivos de vídeo armazenados nos recursos do projeto. A seguir está uma explicação detalhada de como os vídeos foram aplicados:

Carregamento e Exibição de Vídeos:

A função `PlayVideoFromResources(string videoName)` é responsável por iniciar a reprodução de vídeos dentro do aplicativo.

O parâmetro `videoName` determina qual vídeo será reproduzido. Dependendo do valor de `videoName`, o código seleciona o vídeo adequado dos recursos do aplicativo, usando `Resources.cut1` ou `Resources.cut2`.

Armazenamento Temporário de Vídeo:

O vídeo é extraído dos recursos do projeto e armazenado temporariamente no diretório de arquivos temporários do sistema.

Para isso, é gerado um caminho temporário utilizando `Path.GetTempPath()` e um nome único baseado em `Guid.NewGuid()`. O vídeo é gravado neste caminho temporário com a extensão `.mp4`.

Reprodução do Vídeo:

O componente *AxWindowsMediaPlayer* é utilizado para reproduzir o vídeo. O caminho do arquivo temporário gerado é atribuído à propriedade `URL` do *AxWindowsMediaPlayer*.

Após isso, o controle de mídia é configurado para reproduzir o vídeo utilizando o comando `axWindowsMediaPlayer1.Ctlcontrols.play()`.

Tratamento de Erros:

Caso algum erro ocorra durante o processo de carregamento ou reprodução do vídeo, ele é capturado no bloco `catch`, exibindo uma mensagem de erro ao usuário por meio de `MessageBox.Show()`.

```

private void PlayVideoFromResources(string videoName)
{
    videoAtivo = videoName;

    if (Config.Dub == false)
    {
        byte[] video = null;

        switch (videoName)
        {
            case "video":
                video = Resources.cut1;
                lbl_fala2.Visible = true;
                legendaIndex = 8;
                break;
            case "cut2":
                video = Resources.cut2;
                lbl_fala2.Visible = true;
                legendaIndex = 1;

                break;

            default:
                throw new ArgumentException("Nome do video inválido");
        }

        if (video != null)
        {
            try
            {
                string tempFilePath = Path.Combine(Path.GetTempPath(), $"{Guid.NewGuid()}.mp4");

                using (var fs = new FileStream(tempFilePath, FileMode.Create, FileAccess.Write))
                {
                    fs.Write(video, 0, video.Length);
                }

                axWindowsMediaPlayer1.uiMode = "none";
                axWindowsMediaPlayer1.URL = tempFilePath;
                axWindowsMediaPlayer1.Ctlcontrols.play();
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Erro ao reproduzir o vídeo: {ex.Message}");
            }
        }
    }
}

```

Dublagens

14. Como foi feito

Para realizar as dublagens, foi necessário selecionar falas dos personagens no documento “História do jogo - TCC” e deixar a escrita com marcas de oralidade. Para que fosse possível gravar, utilizamos o gravador de áudio do celular. Juntamos o arquivo de áudio com as imagens no aplicativo “CAPCUT” e adicionamos ao jogo, facilitando o processo de inserção de áudio e imagens ao deixá-las juntas.

15. Como foi aplicado

As gravações foram feitas de forma prática, utilizando o gravador de áudio do celular, o que permitiu captar as vozes com qualidade suficiente para o projeto. Após gravar todas as falas, os áudios foram organizados e integrados às imagens correspondentes no aplicativo “CAPCUT”. Esse processo de edição combinou som e imagem, criando arquivos de vídeo prontos para serem utilizados no jogo. A escolha dessa abordagem foi estratégica, pois facilitou a inserção das dublagens ao unir áudio e imagens em um único arquivo.

No jogo, esses vídeos com dublagens foram integrados às cutscenes por meio do método `PlayVideoFromResources()`, que permite carregar e reproduzir vídeos diretamente dos recursos do projeto. Com isso, as cutscenes não só apresentavam as falas dubladas, mas também estavam sincronizadas com as imagens e animações, garantindo uma transição fluida entre a narrativa e a jogabilidade.

```
if (Config.Dub == true)
{
    ReproduzirVideo("fase2intro1Dub");
}

else
{
    ReproduzirVideo("fase2intro1");
}
```

Além disso, para tornar o jogo mais acessível, as dublagens foram complementadas com legendas dinâmicas, criadas com TextBox. Essas legendas permitiram que a narrativa fosse compreendida mesmo por jogadores que optassem por jogar sem som, ampliando a acessibilidade. Essa combinação de elementos resultou em uma narrativa envolvente e inclusiva, que deu destaque ao trabalho de dublagem e aprimorou a experiência geral do jogo.

Fase 1

16. Projeto de fase

A Fase 1 foi planejada com o objetivo de estabelecer uma introdução envolvente e educativa para o jogador, apresentando os elementos principais do enredo, as mecânicas interativas e o tema central do jogo: segurança digital e Phishing. O processo de planejamento levou em conta o equilíbrio entre narrativa, interatividade e aprendizado, para criar uma experiência cativante e significativa.

A narrativa foi construída para inserir o jogador no universo do jogo de forma natural. A história começa com Pítico, o protagonista, em sua rotina diária, permitindo que o jogador se conecte com o personagem antes de ser introduzido ao conflito. Elementos como diálogos com a mãe de Pítico e sua interação inicial com o computador criam um contexto familiar e realista que gradualmente evolui para uma situação inesperada e desafiadora.

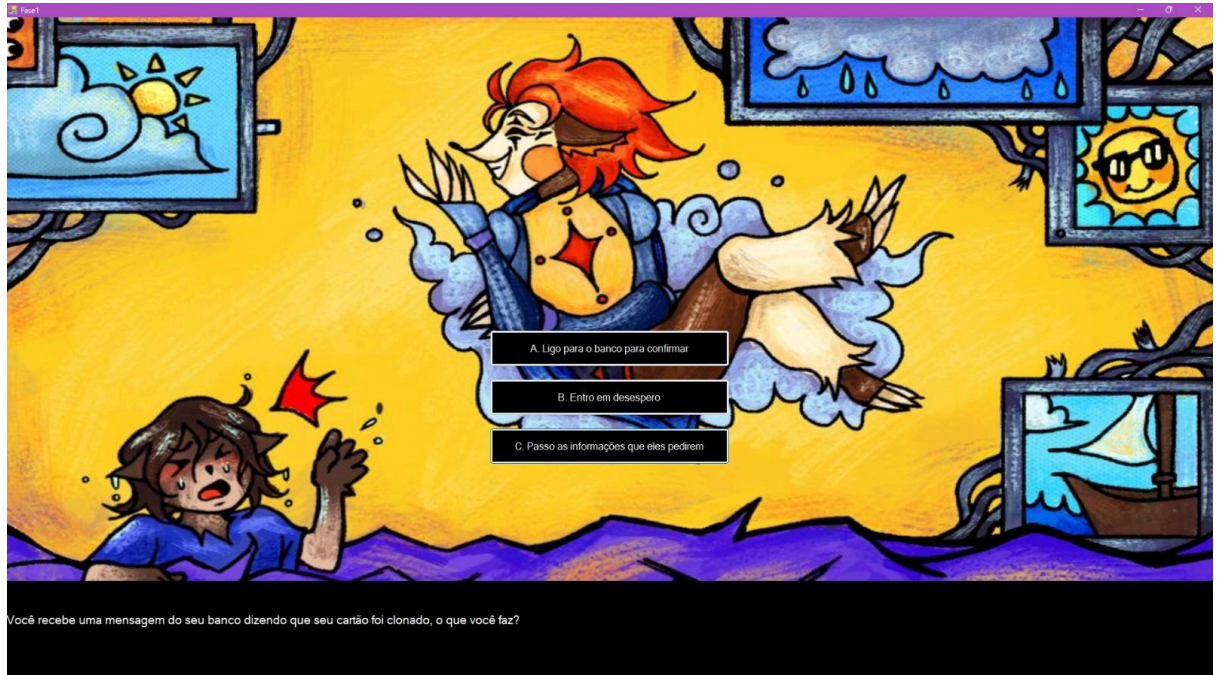
A parte interativa da fase introduz o jogador às mecânicas básicas do jogo, como clicar em elementos na tela e tomar decisões. A sequência em que Pítico interage com anúncios falsos no computador é um ponto-chave para conectar a narrativa com o tema central de Phishing. Após clicar em um link suspeito, o jogador é transportado para um desafio com perguntas e respostas sobre segurança digital, onde as decisões impactam o progresso no jogo.

O desafio de perguntas foi estruturado como uma forma de educar o jogador sobre boas práticas de segurança online. As perguntas apresentam cenários comuns relacionados ao tema e oferecem escolhas que testam o conhecimento do jogador. Para incentivar o aprendizado, o jogador é obrigado a responder corretamente todas as perguntas para progredir, com feedback claro e uma mensagem educativa ao final.

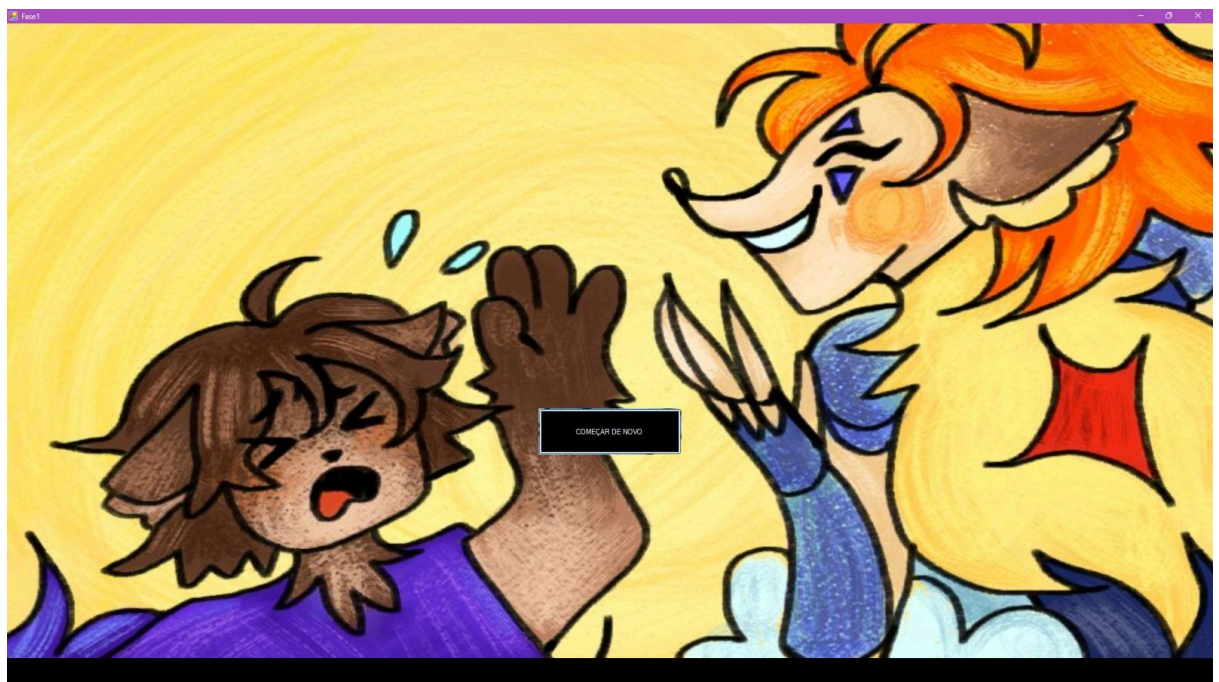
Por fim, a Fase 1 também introduz personagens secundários, como o tatu-bola Pype e os habitantes da vila, que adicionam profundidade à narrativa. Esses personagens ajudam a criar um senso de continuidade e expansão do universo do jogo, incentivando o jogador a explorar mais e se envolver na história. A inclusão de mensagens informativas e educativas ao longo da fase reforça o objetivo do jogo de promover conscientização sobre práticas seguras na internet, enquanto mantém a experiência divertida e dinâmica.

17. Aplicações

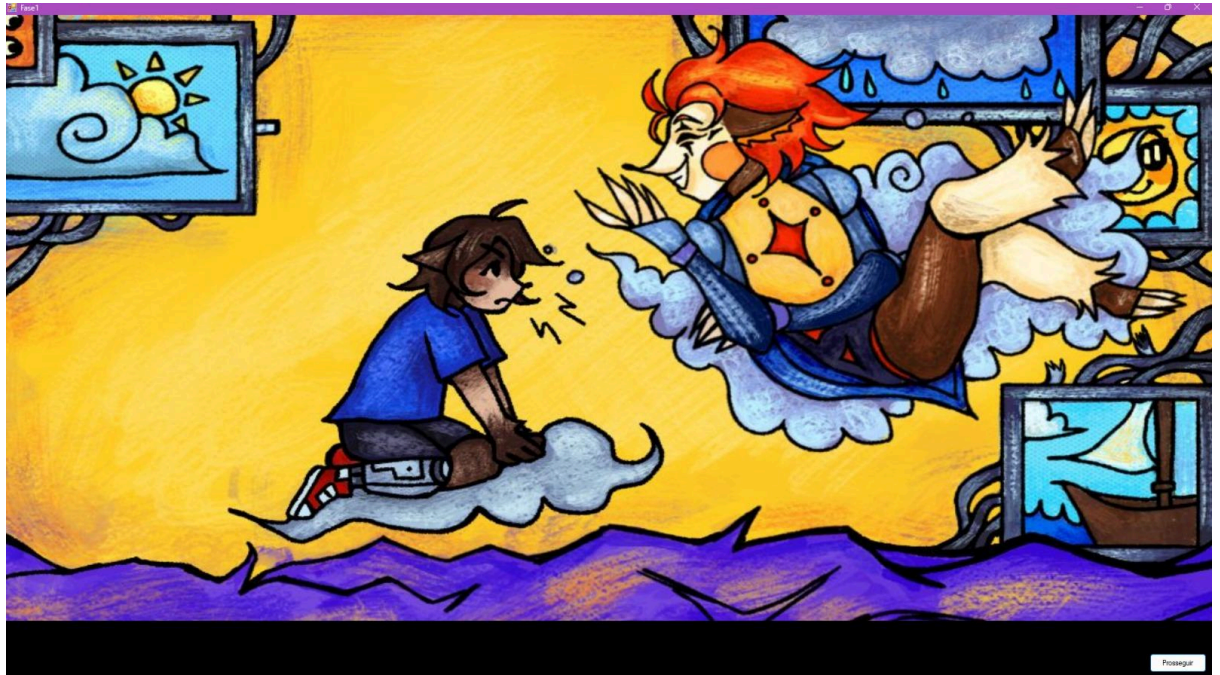
Os botões representam as escolhas da pergunta e foram configurados de uma forma que os textos ficassem na cor branca, com o fundo preto, além de usar o evento CLICK, que é associado a funções específicas que capturam a escolha do jogador e processam a lógica correspondente. Abaixo, o exemplo da pergunta 1.



Ao errar uma questão, o jogador deve retornar a primeira pergunta independente da onde ele esteja, entregando assim uma dificuldade maior a fase. Para isso, foi utilizado um botão em que levava a fase 1.



Utilizamos também a tecla enter e botões para o funcionamento da passagem de cenas, para poder prosseguir a fase. Então, é simplesmente clicar no botão ou teclar ENTER!



Ao responder todas as questões corretamente, você finalmente pode passar para a próxima fase! Para isso, utilizamos um botão que leva as próximas cutscenes.

```
private void prosseguir_fase2_Click(object sender, EventArgs e)
{
    if (prosseguir_fase2.CanSelect & Config.Dub == true)
    {
        button_tentardnv.Visible = false;
        axWindowsMediaPlayer1.Visible = true;
        CarregarVideo(sequenciaVideosDub[indiceVideoAtualDub]);
    }

    else if (prosseguir_fase2.CanSelect & Config.Dub == false)
    {
        button_tentardnv.Visible = false;
        axWindowsMediaPlayer1.Visible = true;
        CarregarVideo(sequenciaVideos[indiceVideoAtual]);
    }
}
```

Também há a possibilidade de refazer a fase 1, caso tenha vontade de jogar novamente. Como forma de parabenização, foi utilizada uma label com o texto “PARABÊNS, VOCÊ PASSOU DA FASE 1!!”



18. Como jogar

Início da fase:

Ao iniciar a Fase 1, você será apresentado ao cenário inicial

O jogo vai lhe apresentar uma pergunta relacionada à situação. No caso da primeira pergunta, será sobre como você reagiria ao saber que seu cartão foi clonado.

Você terá 3 opções de resposta:

A. Ligo para o banco para confirmar.

B. Entro em desespero.

C. Passo as informações que eles pedirem.

Escolha uma opção clicando na resposta desejada. Cada escolha que você fizer irá afetar o resultado do jogo, resultando na sua falha ou no prosseguimento!

Após acertar, pressione a tecla Enter ou clique no botão para avançar para a próxima parte do jogo.

Se sua escolha foi correta, você avançará para a próxima fase do jogo, mas se a escolha não foi a mais adequada, o jogo volta e pede para você tentar novamente, oferecendo uma oportunidade de aprender com seus erros.

Quando você terminar a fase com sucesso, será parabenizado por suas escolhas e com base no desempenho, você será preparado para a próxima fase, onde os desafios se tornam mais complexos.

Fase 2

19. Projeto de fase

A Fase 2 foi planejada para dar continuidade à narrativa iniciada na Fase 1, expandindo a história de Pitico ao introduzir novos conflitos e desafios interativos. O objetivo central dessa fase foi aprofundar o tema do cyberbullying, abordando de forma envolvente questões emocionais e psicológicas, enquanto explora decisões críticas e suas repercussões no universo digital. Além disso, a fase proporciona uma experiência imersiva ao jogador, com novas mecânicas que tornam o combate e a resolução de problemas mais complexos e impactantes.

Conceito e narrativa

A história da Fase 2 segue Pitico, que se encontra em um momento difícil ao ser atacado por mensagens de ódio nas redes sociais. A narrativa gira em torno da maneira como ele lida com essas ofensas, ao lado do sábio Cosenjo, que lhe oferece conselhos sobre como reagir de forma sábia e madura. O tema central da fase é o cyberbullying, explorando como ele afeta emocionalmente os indivíduos e como é importante lidar com isso de forma construtiva. O confronto com Ceshár, o antagonista que personifica a hostilidade virtual, traz uma virada dramática na história, criando uma situação de tensão e aprendizagem para Pitico.

A narrativa foca em situações realistas que o jogador pode encontrar no mundo digital, como ataques anônimos, humilhações públicas e o impacto psicológico dessas agressões. As interações entre Pitico e Cosenjo ajudam a explorar a importância de se proteger online, aprender a lidar com o bullying e buscar alternativas para resolver o problema sem violência.

Design visual e ambientação

Os cenários e elementos visuais da Fase 2 são cuidadosamente projetados para refletir o tom emocional da história. Conforme a narrativa avança e Pitico encontra maneiras de lidar com a situação, a ambientação se torna mais iluminada, simbolizando a superação e o retorno à paz. O uso de animações no fundo e mudanças nas cores do cenário ajudam a expressar as emoções e as reações dos personagens de forma visual, imergindo o jogador no clima da fase.

A figura de Ceshár é representada de maneira imponente e ameaçadora, com um design visual que transmite sua natureza cruel e ameaçadora. Essa caracterização reforça a sensação de opressão que Pitico experimenta durante os ataques de cyberbullying, criando uma atmosfera tensa durante os momentos de confronto.

A Fase 2 introduz novas mecânicas de interação com as decisões do jogador. Pitico deve escolher como reagir aos ataques de Ceshár, com opções que variam de se defender passivamente, como bloquear ou denunciar, até responder com agressão, como xingar de volta. Cada escolha afeta o andamento da batalha e o estado emocional de Pitico, refletindo o impacto real dessas ações no enfrentamento do cyberbullying.

A fase também inclui mecânicas de batalha que acontecem em turnos, onde o jogador deve gerenciar a vida de Pitico, que pode ser afetada pelas ações de Ceshár. O sistema de corações de vida simboliza a saúde emocional de Pitico, que diminui à medida que ele sofre ataques verbais ou toma decisões que o afetam negativamente. Esse sistema cria uma sensação de urgência e faz com que o jogador pense nas consequências de cada escolha.

O foco educativo da Fase 2 está em ensinar o jogador sobre o impacto do cyberbullying e as melhores formas de reagir a ele. As lições abordam:

- Como lidar com ataques anônimos e agressivos online sem reagir com violência;
- A importância de denunciar e bloquear comportamentos abusivos nas redes sociais;
- Como pedir ajuda e buscar apoio social em momentos difíceis.

Esses conceitos são integrados de maneira orgânica à narrativa e às mecânicas do jogo, permitindo que o jogador aprenda enquanto joga, sem que a experiência educativa interfira na diversão e imersão do jogo.

A Fase 2 foi planejada para ser um passo importante na jornada de Pitico, aprofundando os temas do jogo e desafiando o jogador a refletir sobre o impacto emocional do cyberbullying e como reagir de maneira construtiva. Ao combinar narrativa envolvente, mecânicas desafiadoras e elementos educativos, essa fase mantém o equilíbrio entre o aprendizado e a diversão, criando uma experiência imersiva e impactante. O desenvolvimento da história de Pitico e a introdução de novos personagens e escolhas interativas garantem que a fase não apenas expanda o universo do jogo, mas também prepare o terreno para as futuras aventuras e aprendizados do protagonista.

20. Aplicações

Os botões na parte inferior da tela representam as escolhas que o jogador pode fazer diante da situação apresentada. Eles foram implementados para garantir interatividade e um feedback imediato ao jogador.

Descrição:

Pedir Ajuda, Denunciar, Ignorar, Bloquear, Xingar: Cada botão está associado a uma ação específica que reflete a resposta de Pitico à situação.

Design e Posicionamento: Botões com textos claros, dispostos em uma linha horizontal na parte inferior da tela, facilitando o acesso visual e físico.



Função: Quando clicados, os botões acionam eventos específicos no jogo, exibindo o resultado da escolha do jogador.

Cada botão possui um evento `OnClick` vinculado, que registra a escolha e executa a ação correspondente, como:

- Atualizar a narrativa.

- Exibir feedback (mensagem na tela ou animação).

- Alterar o progresso do jogo (avançar ou repetir a fase).

2. Sistema de Feedback Visual

Além dos botões, outros controles visuais foram utilizados para transmitir informações ao jogador:

Barra de Corações (Topo da Tela):



Representa a "vida" de Pitico.

Diminui, aumenta ou permanece constante dependendo da escolha feita pelo jogador, refletindo as consequências das ações.

Texto na Parte Inferior ("O que o Pitico fará?"):



Implementado como um Label, serve como uma orientação narrativa, guiando o jogador a tomar uma decisão.

Sprites dos Personagens:

Pitico (azul): Representado como o personagem principal, posicionado à esquerda.

CSHAR (roxo): Colocado em destaque na direita, representando a situação desafiadora.

Ambos foram gerenciados por PictureBox, garantindo fácil manipulação e visibilidade.



3. Configurações de Fluxo da Fase

A fase utiliza uma lógica condicional que é acionada com base nas interações do jogador:

Sequência de Eventos:

O jogador escolhe uma das opções (botões).

Um evento avalia a escolha e aciona a resposta apropriada:

Escolhas corretas: Feedback positivo, continuidade da narrativa.

Escolhas erradas: Feedback negativo, possibilidade de tentar novamente.

Isso foi implementado por meio de blocos if-else ou switch, avaliando a decisão em tempo

real.


```
else if (botao == btn_denunciar)
{
    pitico_1.Visible = false;
    pitico_2.Visible = false;
    pitico_3.Visible = true;
    VidaCshar1.BringToFront();
    VidaCshar2.BringToFront();
    VidaCshar3.BringToFront();
    VidaCshar4.BringToFront();
    VidaCshar5.BringToFront();
    VidaPitico1.BringToFront();
    VidaPitico2.BringToFront();
    VidaPitico3.BringToFront();
    VidaPitico4.BringToFront();
    VidaPitico5.BringToFront();
    label1.BringToFront();
    //pitico_3.BringToFront();
    cshar_1.Visible = false;
    cshar_2.Visible = true;
    cshar_3.Visible = false;
    //cshar_2.BringToFront();
    vidaCshar--;
    if (vidaCshar < 0) vidaCshar = 0;
}
```

21. Como jogar

O jogo apresenta Pitico diante de um personagem ameaçador que simboliza práticas de assédio online (CSHAR)

A mensagem principal aparece no inferior da tela: "O que o Pitico fará?"

Essa pergunta define o propósito da fase, desafiando o jogador a tomar uma decisão sobre como lidar com a situação.

Cinco opções de resposta aparecem na parte inferior da tela. São elas:

Pedir ajuda: Pitico solicita auxílio de uma autoridade ou de amigos confiáveis.

Denunciar: Escolher denunciar o comportamento do CSHAR na plataforma ou a uma autoridade.

Ignorar: Não interagir e seguir em frente.

Bloquear: Impedir qualquer contato futuro.

Xingar: Responder de forma agressiva ao comportamento ofensivo.

Para escolher uma opção, clique diretamente no botão correspondente. Cada decisão leva a uma consequência. Dependendo da escolha, uma vida será perdida, do jogador ou do adversário.

Assim que uma opção é selecionada, o jogo exibe uma animação ou texto explicativo mostrando os impactos da escolha..

Mensagens educativas destacam a importância de reportar e lidar com situações desse tipo de forma madura e assertiva.

Cada personagem possui 5 vidas, e após o término dessas 5 vidas, se o jogador vencer, a fase prossegue. Caso perca, a fase poderá ser reiniciada.

O objetivo é aprender a identificar comportamentos problemáticos e a usar ferramentas digitais para combatê-los de maneira responsável.

Fase 3

22. Projeto de fase

Na Fase 3, Pitico se encontra em uma nova situação cheia de desafios digitais, após ter desmaiado ao final da batalha anterior. Levado por um misterioso morador da vila, ele logo encontra aliados inesperados e é chamado para proteger a vila de uma ameaça de malware crescente. O objetivo central desta fase é enfrentar 10 tipos de vírus diferentes e derrotá-los sem perder todas as suas vidas. O jogador precisará usar a tecla ****Espaço**** para ativar o antivírus e derrotar os invasores, ao mesmo tempo em que protege Pitico de perder vida a cada toque dos vírus.

A história da Fase 3 se aprofunda no conceito de segurança digital. Após Pitico desmaiar devido à sua luta anterior, ele é resgatado por Tio Edu, um leopardo sábio, e logo se une a novos aliados: Thiago, o ornitorrinco, e Augusto, a tartaruga. Juntos, eles descobrem que a vila está sendo atacada por vírus, e Pitico, sendo o herói da história, precisa usar seus conhecimentos para enfrentar as ameaças e proteger os moradores. A narrativa mistura momentos de aprendizado com ação intensa, à medida que Pitico encara diversos tipos de malware que estão infectando a vila.

Durante essa fase, Pitico aprende sobre diferentes tipos de malwares, como Spyware, Trojan, Ransomware, e Scareware. Os aliados de Pitico, Thiago e Augusto, fornecem dicas sobre como usar o antivírus e encorajam o jovem herói a se preparar para a batalha. O objetivo da fase é sobreviver aos ataques e derrotar os vírus, aprendendo, ao mesmo tempo, como evitar e remover malwares no mundo real.

A figura de Pitico, acompanhado por seus novos amigos, caminha pela vila, enfrentando os ataques dos malwares. À medida que os vírus são derrotados, a vila começa a se recuperar, mostrando a importância de proteger o mundo digital contra essas ameaças. O uso de animações de combate, como os efeitos do antivírus e a movimentação dos vírus, cria uma atmosfera de tensão e urgência.

A Fase 3 traz mecânicas de combate mais dinâmicas, com Pitico enfrentando diferentes tipos de vírus em um combate rápido. O jogador precisa usar a tecla ****Espaço**** para ativar o antivírus quando um vírus se aproximar de Pitico. Cada tipo de vírus causa diferentes níveis de dano, e o jogador deve atacar rapidamente para evitar que Pitico perca todas as suas vidas.

Os vírus enfrentados incluem:

- ****Spyware (-1 de vida)****: O mais comum e invisível, normalmente encontrado em redes sociais e aplicativos.
- ****Trojan (-2 de vida)****: Um vírus disfarçado que se instala ao clicar em links suspeitos ou fazer downloads duvidosos.
- ****Ransomware (-3 de vida)****: Um vírus que criptografa arquivos e exige pagamento para a recuperação.
- ****Scareware (-4 de vida)****: Um malware que engana o usuário, levando-o a sites perigosos ou fazendo-o comprar aplicativos falsos.

Para vencer, o jogador deve derrotar todos os vírus antes que Pitico perca todas as suas vidas. Cada vez que um vírus é derrotado, o jogador avança para o próximo inimigo. As batalhas exigem rapidez e estratégia para escolher o momento certo de usar o antivírus, enquanto evita os danos causados pelos vírus.

****Caso perca a batalha****

- "Droga... eles são muitos."

(Cutscene de game over de Pitico, com a vila destruída e Pitico caindo de cansaço.)

****Caso vença a batalha****

- "Meu Deus... Não acredito que finalmente acabou."

(Cutscene mostrando Pitico cansado, com suor na testa, olhando para os vírus derrotados ao fundo.)

23. Aplicações

Movimentação do personagem e interação com inimigos:
O personagem Pitico se movimenta usando as teclas W, A, S, D (cima, esquerda, baixo, direita).

```
private void Movimento_Pitico(object sender, KeyEventArgs e)
{
    int moveStep = 10; // Define a quantidade de pixels para cada movimento

    switch (e.KeyCode)
    {
        case Keys.W: // Movimenta para cima
            Pitico_walk.Top -= moveStep;
            Pitico_walk.Image = Properties.Resources.pitico_andando_de_costas_1;
            break;

        case Keys.A: // Movimenta para a esquerda
            Pitico_walk.Left -= moveStep;
            Pitico_walk.Image = Properties.Resources.pitico_andando_pra_esquerda_1;
            break;

        case Keys.S: // Movimenta para baixo
            Pitico_walk.Top += moveStep;
            Pitico_walk.Image = Properties.Resources.pitico_walk_1;
            break;

        case Keys.D: // Movimenta para a direita
            Pitico_walk.Left += moveStep;
            Pitico_walk.Image = Properties.Resources.pitico_andando_pra_direita_1;
            break;
    }
}
```

Quando o Pitico colide com o "Spyware", uma função chamada EmpurrarPersonagem é ativada, que reduz a vida do personagem e aplica um "empurrão", deslocando o personagem quando ele colide com o "Spyware".

O movimento do "Spyware" é controlado por um temporizador (timerSpyware), que faz com que o "Spyware" siga o Pitico em direção à sua posição.

```

private void EmpurrarPersonagem(KeyEventArgs e, Control spyware)
{
    int deslizeVelocidade = 50;

    if (Pitico_walk.Bounds.Intersects(spyware.Bounds))
    {
        vida--;
        if (Pitico_walk.Left < spyware.Left)
            Pitico_walk.Left -= deslizeVelocidade;
        else if (Pitico_walk.Left > spyware.Left)
            Pitico_walk.Left += deslizeVelocidade;

        if (Pitico_walk.Top < spyware.Top)
            Pitico_walk.Top -= deslizeVelocidade;
        else if (Pitico_walk.Top > spyware.Top)
            Pitico_walk.Top += deslizeVelocidade;
    }
}

```

```

}
private void TimerSpyware_Tick(object sender, EventArgs e)
{
    MoverSpyware();
}

```

```

private void MoverSpyware()
{
    int movimentoX = 0;
    int movimentoY = 0;

    if (Pitico_walk.Left > Spyware.Left)
    {
        movimentoX = 1;
    }
    else if (Pitico_walk.Left < Spyware.Left)
    {
        movimentoX = -1;
    }

    if (Pitico_walk.Top > Spyware.Top)
    {
        movimentoY = 1;
    }
    else if (Pitico_walk.Top < Spyware.Top)
    {
        movimentoY = -1;
    }

    int velocidade = 2;

    // Atualizando a posição do Spyware
    Spyware.Left += movimentoX * velocidade;
    Spyware.Top += movimentoY * velocidade;

    if (Spyware.Bounds.Intersects(Pitico_walk.Bounds))
    {
        EmpurrarPersonagem(null, Spyware);
    }
}

```

Gerenciamento de vidas:

O jogo tem um sistema de vidas. A cada colisão com o "Spyware", a vida do Pitico é decrementada.

O número de vidas restantes é visualizado na interface com o uso de PictureBox (os elementos VidaPitico1, VidaPitico2, etc.).

Quando a vida chega a zero, o jogo exibe uma mensagem de "Você Perdeu!".

```
if (vida == 5)
{
    VidaPitico5.Visible = true;
    VidaPitico4.Visible = true;
    VidaPitico3.Visible = true;
    VidaPitico2.Visible = true;
    VidaPitico1.Visible = true;
}

if (vida == 4)
{
    VidaPitico5.Visible = false;
    VidaPitico4.Visible = true;
    VidaPitico3.Visible = true;
    VidaPitico2.Visible = true;
    VidaPitico1.Visible = true;
}

if (vida == 3)
{
    VidaPitico5.Visible = false;
    VidaPitico4.Visible = false;
    VidaPitico3.Visible = true;
    VidaPitico2.Visible = true;
    VidaPitico1.Visible = true;
}

if (vida == 2)
{
    VidaPitico5.Visible = false;
    VidaPitico4.Visible = false;
    VidaPitico3.Visible = false;
    VidaPitico2.Visible = true;
    VidaPitico1.Visible = true;
}

if (vida == 1)
{
    VidaPitico5.Visible = false;
    VidaPitico4.Visible = false;
    VidaPitico3.Visible = false;
    VidaPitico2.Visible = false;
    VidaPitico1.Visible = true;
}

if (vida == 0)
{
    VidaPitico5.Visible = false;
    VidaPitico4.Visible = false;
    VidaPitico3.Visible = false;
    VidaPitico2.Visible = false;
    VidaPitico1.Visible = false;

    MessageBox.Show("Você Perdeu!");
}
}
```

O jogo inclui vídeos que são reproduzidos usando o componente AxWindowsMediaPlayer. O código fornece um método ReproduzirVideo que carrega um vídeo a partir dos recursos embutidos e o reproduz.

Há também um controle de troca de vídeos com o método `AdvanceVideo`, que alterna entre diferentes vídeos dependendo do estado atual do jogo.

O cenário de fundo pode ser alterado (com o método `TrocarCenario`) quando o personagem atinge certas coordenadas.

```
private void TrocarCenario()
{
    this.BackgroundImage = null;

    this.BackgroundImage = Properties.Resources.cenário_com_as_casas_2;

    Pitico_walk.Location = new Point(0, 360);
}
```

Ajustes de tela e resolução:

O código ajusta a resolução da janela do jogo de acordo com a proporção de 16:9, ajustando o tamanho da tela e dos vídeos para que o jogo seja exibido corretamente em diferentes resoluções de tela.

Ao final de certos vídeos, o código esconde o player de vídeo ou avança para o próximo vídeo, conforme a lógica definida em `AdvanceVideo`.

O jogo usa o evento `KeyDown` para capturar as teclas pressionadas e controlar a movimentação do Pitico, além de outras ações (como avançar o vídeo com a tecla `Enter`).

24. Como jogar

Movimentação:

Use as teclas `W`, `A`, `S` e `D` para mover o personagem Pitico para cima, para a esquerda, para baixo e para a direita, respectivamente.

O personagem Pitico se move de acordo com as teclas pressionadas e a animação da imagem será ajustada conforme a direção.

Objetivo:

O objetivo do jogo é derrotar os Vírus (os inimigos) que perseguem Pitico na tela.

Para derrotar o Vírus", pressione a tecla Espaço quando Pitico estiver próximo do inimigo. Isso irá causar um "ataque" no "Vírus", derrotando-o.

Proximidade do Inimigo:

O Vírus se move em direção ao Pitico. Quando Pitico estiver perto o suficiente, pressionar a tecla Espaço vai fazer com que o Pitico ataque o Vírus.

Se o ataque for bem-sucedido, o Vírus será removido da tela e você poderá continuar avançando no jogo.

Vida:

Pitico começa com 5 vidas. Cada vez que o Vírus colidir com o personagem, uma vida será perdida.

Se a vida chegar a 0, o jogo termina e você perde.

Vencer o Jogo:

Para vencer, continue derrotando o Vírus e evite ser atingido por ele.

O jogo termina quando o Vírus for derrotado ou quando Pitico perder todas as suas vidas.