

Sesc escola curso de programação de jogos digitais

NAthalia amorim abade

Cuiabá

2024

Nathalia amorim abade

integração dos elementos multimídia ao jogo digital

apresentada ao Curso de Programação de Jogos Digitais, Sesc Escola.

Discente: Wanderson

Docente: Nathalia Amorim Abade

cuiabá

2024

Sumário

Resumo 4

1 importação de Assets 5

1.1 formatos e compatibilidade 5

1.2 modelos e sprites 6

2 repositórios online de artes

2.1 obtenção de assets

2.2 importação e utilização.

3 animação em engine

3.1 criação e manipulação

3.2 configuração e controle via código

4 manipulação de mapas de imagens

4.1 configurações de mapas spriteshetse de tilemaps

5 áudio em engine

5.1 manipulação e controle

5.2 componentes para áudio e controle via código

5.3 padrão singleton e práticas no gerenciamento de áudio

6 materiais e texturas

6.1 conceitos, configuração e aplicação

7 sistema de partículas

7.1 conceito e utilização

8 controle de versões do projeto

8.1 recurso de nuvem, interação com engine e configuração do repositório

9 validação da integração

9.1 procedimentos e técnicas de testes

10 programação orientada a objetos

10.1 estruturas de dados e tipos genéricos

ReferÊncias

**RESUMO**

INTEGRAÇÃO DOS ELEMENTOS MULTIMÍDIA AO JOGO DIGITAL

**Objetivo:** Objetivo da integração dos elementos multimídia aos jogos digitais é utilizar de diversos meios de comunicação, assim tornando acessível e possibilitando elaborar inúmeros jogos, sejam eles: Sensoriais, recreativos, competitivos, mentais,  educacionais, etc

# 1 IMPORTAÇÃO DE ASSETS

## FORMATOS E COMPATIBILIDADE.

A Unity suporta assets criados por diversas ferramentas, você pode editar seus assets a qualquer momento, e instamtameamente ver os resultados no jogo.

Exemplos de arquivos:

Arquivos 3D suportados:

* Maya .mb e .ma
* 3D Studio Max .max
* Cheetah 3D .jas
* Cinema 4D .c4d
* Blender .blend
* modo .lxo
* Autodesk FBX
* COLLADA
* Carrara
* Lightwave
* XSI 5.x
* SketchUp Pro
* Wings 3D
* 3D Studio .3ds
* Wavefront .obj
* Drawing Interchange Files .dxf

Arquivos de imagens suportados:

* Photoshop .psd e .tiff
* JPEG
* PNG
* GIF
* BMP
* TGA
* IFF
* PICT

Arquivos de áudio e vídeo suportados

* MP3, OGG, WAV, …
* MOV, AVI, ASF, MPG, MPEG, MP4

## **modelos e sprites**

Um sprite é um personagem ou um objeto dentro dos jogos 2D. Para criar o efeito de movimento, são necessários vários sprites diferentes com pequenas variações do mesmo personagem.

# 2. REPOSITÓRIOS ONLINE DE ARTES

## **OBTENÇÃO DE ASSESts**

Em lojas de assets como a Unity Asset Store é possível baixar itens diversos para compor as partes de um jogo – que vão desde a arte gráfica até a parte de programação.

## **importação e utilização**

existe um botão chamado “Import” o qual permitirá que você importe esse asset para o projeto. Caso esteja baixando esse asset pela primeira vez, o botão se chamará “Download” e, nas vezes seguintes, terá o nome de “Import”.

O asset é utilizado de diversa maneira na criação de jogos, pois trata-se de uma ferramenta fundamental. Ele representa os elementos que constituem um jogo, como

ilustrações, efeitos sonoros, scripts, etc.

3. animação em engine

Para criar e manipular animações na Unity, você seguirá um processo que aproveita as ferramentas integradas da Unity, como o Animator e o Animation Clip Editor. Aqui está um guia passo a passo focado na Unity:

1. Preparando seus Assets

Se você está trabalhando com animações 3D, certifique-se de que seus modelos 3D já estão com rigging e possivelmente com algumas animações pré-feitas que foram exportadas em um formato compatível com a Unity (como .fbx). Para animações 2D, você deve ter seus sprites prontos em arquivos de imagem.

2. Criando Animações na Unity

Para Animações 2D:

1. Selecionar o Sprite: No painel Project, selecione o sprite que você deseja animar.

2. Abrir o Animation Window: Vá para Window > Animation > Animation para abrir a janela de animação.

3. Criar um Novo Clip de Animação: Com o GameObject que você quer animar selecionado, clique no botão "Create" na janela Animation para criar um novo Animation Clip. Salve o clip em sua pasta Assets.

4. Adicionar Frames: Arraste e solte seus sprites na timeline do Animation Clip. Ajuste os frames conforme necessário para criar a sequência de animação desejada.

Para Animações 3D:

1. Configurar o Animator: Seu modelo 3D importado deve já ter um componente Animator. Se não, adicione manualmente ao GameObject.

2. Criar uma Animator Controller: No painel Project, clique com o botão direito e escolha Create > Animator Controller. Nomeie e abra este controller.

3. Vincular o Animator Controller ao seu GameObject: Selecione o GameObject e, no componente Animator, arraste o Animator Controller criado para o campo 'Controller'.

4. Configurar Estados e Transições: Use a janela Animator (Window > Animation > Animator) para configurar os estados de animação e as transições entre eles, arrastando seus clips de animação para o canvas do Animator.

**4.MANIPULAÇÃO DE MAPA DE IMAGENS**

A manipulação de mapas de imagens em uma engine de jogos é o processo de criar e utilizar imagens para diversos fins, como texturas de terreno, superfícies de objetos e efeitos visuais. Isso é fundamental para criar ambientes e cenários detalhados e realistas dentro do jogo.

**4.1**  
A configuração de sprite de tilemaps em uma engine de jogos é o processo de organizar sprites para representar diferentes elementos do ambiente, como solo, paredes e objetos. Esses sprites são então utilizados para construir os níveis do jogo, permitindo uma criação eficiente e flexível de cenários.

**5.ÁUDIO EM ENGINE**

Áudio em uma engine de jogos se refere ao uso de recursos sonoros, como efeitos sonoros, trilhas sonoras e diálogos, para criar uma experiência imersiva e envolvente para o jogador. Esses elementos são integrados ao jogo para aumentar a atmosfera e a interatividade.

**5.1** Configurar componentes de áudio em uma engine de jogos envolve definir e ajustar elementos sonoros, como efeitos sonoros e trilhas sonoras, para criar uma experiência sonora adequada ao jogo. Isso pode incluir o controle de volume, reprodução, pausa e outras propriedades dos áudios por meio de código.

**5.2**

Animação em uma engine de jogos é o processo de adicionar movimento a objetos e personagens, tornando o jogo mais dinâmico e realista. Isso é feito definindo diferentes estados ou poses ao longo do tempo para criar uma sequência de movimento fluente.

**6.MATERIAIS E TEXTURA**

Materiais e texturas em uma engine de jogos são usados para definir a aparência visual dos objetos e cenários. As texturas são imagens aplicadas à superfície dos modelos 3D para fornecer detalhes, enquanto os materiais controlam como a luz interage com essas texturas. Juntos, eles criam ambientes visualmente ricos e realistas no jogo.

**6,1** Conceitos em uma engine de jogos referem-se a ideias fundamentais e princípios que guiam o desenvolvimento e funcionamento do jogo. Isso pode incluir conceitos relacionados à física, renderização gráfica, inteligência artificial, lógica de jogo, otimização de desempenho, entre outros. Compreender esses conceitos é essencial para criar jogos eficientes, imersivos e divertidos.

**6.**2Configurações e aplicações em uma engine de jogos referem-se à definição e uso de parâmetros e recursos para personalizar e controlar vários aspectos do jogo. Isso inclui ajustes relacionados a gráficos, áudio, física e outros elementos, garantindo uma experiência de jogo personalizada e envolvente**.**

**7. SISTEMA DE PARTÍCULAS**

Um sistema de partículas em uma engine de jogos é uma técnica utilizada para simular e renderizar uma grande quantidade de elementos gráficos chamados partículas. Essas partículas podem representar efeitos visuais como fogo, fumaça, explosões, chuva, entre outros, adicionando detalhes e realismo ao jogo.

**7.1**

Sistemas de partículas em uma engine de jogos são utilizados para criar efeitos visuais como fogo, fumaça, explosões e chuva, adicionando detalhes e realismo aos jogos. Eles são controlados por parâmetros como posição, velocidade e cor, permitindo aos desenvolvedores criar ambientes imersivos e dinâmicos.

**8.CONTROLE DE VERSÕES**

O controle de versão de um projeto de jogo digital envolve gerenciar as diferentes versões e alterações feitas ao longo do desenvolvimento usando ferramentas como Git ou SVN. Isso permite acompanhar as modificações, comparar versões e colaborar de forma eficiente em equipe.

**8,1**

Recursos na nuvem são serviços e capacidades computacionais disponíveis pela internet, como armazenamento de dados e hospedagem de aplicativos, oferecendo flexibilidade e acessibilidade para usuários e organizações.

**8.2**

A integração com uma engine de jogos e a configuração de um repositório envolvem o uso de ferramentas de controle de versão, como Git, para rastrear e gerenciar alterações no projeto. Isso permite colaboração entre a equipe, mantendo um histórico de mudanças e facilitando a sincronização. A integração com a engine pode incluir plugins ou scripts para automação de tarefas relacionadas ao controle de versão.

**9.VALIDAÇÃO DA INTEGRAÇÃO**

A validação da integração envolve verificar se a comunicação entre a engine de jogos e o repositório está funcionando corretamente. Isso pode ser feito testando o fluxo de trabalho de controle de versão, como fazer check-in e check-out de arquivos, sincronizar alterações entre diferentes membros da equipe e garantir que as atualizações do projeto sejam refletidas corretamente no repositório. Além disso, é importante verificar se os plugins ou scripts de integração estão configurados corretamente e se estão funcionando conforme o esperado dentro da engine de jogos. Essa validação garante que a integração esteja pronta para uso e que a equipe possa colaborar efetivamente no desenvolvimento do jogo.

**9.1**

Procedimentos e técnicas de teste em um jogo digital envolvem verificar a funcionalidade, desempenho, usabilidade, compatibilidade, segurança e localização do jogo. Isso garante uma experiência de jogo suave e satisfatória para os jogadores.

**10. PROGRAMAÇÃO ORIENTADA A OBJETOS**

Programação Orientada a Objetos (POO) é um paradigma de programação que organiza o código em torno de "objetos", que podem conter dados (atributos) e comportamentos (métodos). Isso promove a reutilização de código, facilita a manutenção e permite uma modelagem mais próxima do mundo real.

**10.1**

Estrutura de dados é a maneira como os dados são organizados em um programa, facilitando sua manipulação e armazenamento. Tipos genéricos são uma abstração na programação que permitem escrever código que pode trabalhar com qualquer tipo de dado, oferecendo flexibilidade e reutilização de código.

Parte superior do formulário

ReferÊncias

(Produção de Jogos) -

https://producaodejogos.com/game-assets-que-ajudam-no-desenvolvimento-do-seu-jogo/#

(Alura) -

https://www.alura.com.br/artigos/sprite-como-funciona-em-jogos-2d

(Chat GPT)