unity

unity

Certified

Expert

Technical Artist:
Shading & Effects

Objetivos do Exame

Especialista Certificado Unity
Artista Técnico:
Sombreamento & Efeitos

Função

O Artista Técnico de Sombreamento e Efeitos se concentra em entregar a concepção visual por trás do jogo. Artistas com habilidades e competência em sombreamento e efeitos costumam trabalhar com outros Artistas Técnicos e de Efeitos para preparar assets ou melhorar assets já preparados. Artistas de Sombreamento e Efeitos são responsáveis por implementar o visual, estilo, tema e estética do jogo.

Artistas com essas habilidades essenciais implementam iluminação bake e em tempo real para criar e personalizar shaders e sistemas de renderização, além de criar particle systems e efeitos que interagem com outros assets.

Títulos de trabalho para esta função

- Escritor de Shader
- Lighter
- Artista de Efeitos

Pré-requisitos

Esta certificação de especialista é recomendada para pessoas trabalhando há muitos anos neste ramo, com grande experiência em aplicação prática avançada, por exemplo:

- Experiência em um estúdio de desenvolvimento de jogos, com pelo menos dois títulos lançados
- Grande conhecimento de técnicas e workflows de iluminação baseada em física
- Compreensão completa de criação de material para pipelines de renderização baseados em física
- Compreensão completa de correção de cor e post effects
- Grande conhecimento de conceitos de fotografia
- Experiência em criar shaders em HLSL, CgFX ou outras linguagens
- Conhecimento de scripting e programação em linguagens como C++, C# ou Unityscript
- Grande conhecimento de particle systems, simulações dinâmicas e formatos de intercâmbio, como Alembic
- Domínio de ferramentas de criação de assets, como Adobe Creative Suite, Substance Designer, Substance Painter, Quixel Suite, etc.
- Grande compreensão de conceitos matemáticos 2D e 3D

Habilidades essenciais

A certificação de Artista Técnico Especialista: Sombreamento & Efeitos verifica que os candidatos tenham as habilidades necessárias para implementar de forma eficiente o visual, estilo, tema e estética do jogo. Os candidatos bem sucedidos têm conhecimento avançado nas seguintes áreas.

Prototipagem

- Criar e avaliar protótipos de shaders e de materiais

Shaders e Materiais

- Construir e testar shaders personalizados para:
    - Simular fenômenos
    - Responder a eventos do jogo dinamicamente
    - Ampliar a funcionalidade de shaders padrões para dar suporte ao workflow do desenvolvimento visual
    - Implementar modelos personalizados de iluminação e visuais não-fotorrealistas (NPR)
- Projetar, construir e implementar materiais procedurais e efeitos de materiais que se adaptam ao design e dados de entrada da cena
- Implementar material UI personalizado usando ShaderGUI
- Criar Inspectors personalizados usando OnInspectorGUI()
- Implementar post-effects (por exemplo, depth of field, correção de cor, bloom, screen space reflections, motion blur e Fog) para corresponder à cinematografia específica presente no GDD
- Fazer script do uso de Render Textures para gerenciar reflexos em tempo real

Renderização e Iluminação

- Entender as diferenças entre diferentes tipos de luzes e seus impactos no desempenho
- Entender as diferenças entre diferentes tipos de sombras e seus impactos no desempenho
- Entender a diferença de renderização entre Forward e Deferred
- Determinar os requisitos de renderização de API e suas restrições de acordo com a plataforma
- Adaptar e ampliar a pipeline de renderização usando a API Unity, command buffers e a biblioteca Gráfica

Particle Systems

- Simular fenômenos atmosféricos usando múltiplos Particle Systems
- Implementar efeitos típicos de jogos, como fogo, explosões, fumaça e água
- Criar efeitos de partículas complexos incluindo Particle Systems com Sub-Emitters, Line e Trail Renderers
- Fazer script de eventos de Particle Systems para acontecerem durante o jogo como resposta a comportamentos do jogador, de NPCs ou de outros eventos em tempo de execução.
- Importar e renderizar dados de simulação gerados externamente
- Avaliar dinamicamente dados do Collider e do Transform para implementar interações com Particle Systems

Desempenho e Otimização

- Entender as especificações e limitações da plataforma alvo
- Otimizar shaders, Particle Systems, post effects, iluminação, Fog, sombras, etc. para rodarem na plataforma alvo
- Saber quando usar técnicas de otimização e resolução de problemas (billboarding, problemas com alpha sorting, draw calls, problemas com fill-rate, cenários ligados à CPU/GPU) onde for necessário
- Analisar e avaliar problemas de renderização com o Frame Debugger e ferramentas de captura específicas de cada plataforma

Tópicos do Exame de Certificação

Tooling e Pipeline

- Personalização de asset
- Processar melhorias através de ferramentas personalizadas e personalização do Editor

Renderização

- Pipeline de renderização
- Efeitos de pós-processamento
- Câmeras na Unity

Shaders

- Construção de shaders, prototipagem e personalização
- Conhecimento do shader Render Setup
- Conhecimento de scripting relativo a shaders

Partículas e Efeitos

- Personalização e ampliação de Particle Systems
- Técnicas de efeitos

Desempenho

- Otimização de cena

Exemplos de questões

Questão 1

Um Programador de Jogabilidade está criando um protótipo de um jogo de tiro side scrolling. O personagem principal é um avião e os inimigos são tipos diferentes de Óvnis. Quando o jogador destrói um Óvni, um efeito de explosão é exibido no lugar do Óvni. Quando o jogador morre, um efeito de explosão especial da nave do jogador é exibido.

O avião pode disparar até 64 tiros na tela. O número máximo de Óvnis de qualquer tipo exibido na tela é 128. O número máximo de tiros de Óvnis na tela é 1024. O número máximo de efeitos de explosão na tela é 128.

O Programador precisa determinar quais GameObjects devem ser colocados no editor e quais devem ser gerados em tempo de execução (por exemplo, de um pool de objetos).

Como isso deve ser feito?

A O jogador, os Óvnis, o efeito de explosão do jogador e dos Óvnis devem ser colocados no editor. Os tiros do jogador e dos Óvnis devem ser gerados em tempo de execução.

B O jogador, os tiros do jogador, os Óvnis e os tiros dos Óvnis devem ser colocados no editor. O efeito de explosão do jogador e dos Óvnis devem ser gerados em tempo de execução.

C O jogador e os Óvnis devem ser colocados no editor. Os tiros do jogador e dos Óvnis e os efeitos de explosão do jogador e dos Óvnis devem ser gerados em tempo de execução.

D O jogador e o efeito de explosão do jogador devem ser colocados no editor. Os Óvnis, os tiros dos Óvnis e do jogador e o efeito de explosão dos Óvnis devem ser gerados em tempo de execução.

Questão 2

O Game Design Document (GDD) é um jogo de mundo aberto em terceira pessoa. O GDD especifica dois tipos de jogabilidade para o jogador principal:

1. Andar pelo mundo a pé
2. Dirigir uma motocicleta

A altura relativa de cada modo é similar, mas o jogador vai muito mais rápido com a motocicleta do que a pé.

O Programador de Jogabilidade determina diversas áreas do jogo que devem ser ajustadas durante a troca de andar para pilotar.

- Field of View (FOV) da câmera
- Distâncias de Level of Details (LOD)
- Level streaming

Quando o jogador está na motocicleta, qual estratégia o Programador deve usar para lidar com cada área neste GDD?

A O FOV da câmera deve ser um valor menor.
O level streaming deve ser muito mais rápido.

A O FOV da câmera deve ser um valor maior.
O level streaming deve ser muito mais lento.

C As distâncias de LOD devem ser um valor maior.
O level streaming deve ser muito mais lento.

D O FOV da câmera deve ser um valor maior.
O level streaming deve ser muito mais rápido.

Questão 3

O GDD se passa em uma grande cidade. O jogador pode andar por ela e interagir com qualquer um. O jogador pode concluir várias missões aleatórias para os cidadãos. O jogador pode ter apenas cinco missões ativas por vez. Para andar de uma ponta da cidade para a outra, o jogador precisa de aproximadamente quatro horas.

O jogo contém um minimapa visto de cima no Heads-Up Display (HUD) que mostra um ícone 2D de cada um dos seguintes:

- localização do jogador
- todas as missões ativas
- todos os cidadãos andando pela cidade

É possível aumentar e diminuir o zoom no minimapa. No zoom máximo, apenas o ícone do jogador é visível. No zoom mínimo, 25% da cidade é visível. Há uma lentidão que acontece apenas quando o jogador aumenta o zoom no minimapa ao máximo.

Qual é uma possível causa para essa lentidão?

A Há muitas requisições para descarregar todas as texturas de ícones dos cidadãos da memória.

B Os ícones de missões ativas estão sendo consultados com muita frequência para saber quantos estão ativos.

C O algoritmo de culling do minimapa está levando muito tempo para remover grande parte da cidade.

D O HUD é muito complicado e está sendo recriado a cada quadro.

Questão 4

O GDD é um jogo para dispositivos móveis com pouca memória disponível. O jogo tem seções carregadas em AssetBundles para garantir um bom desempenho. Cada seção é independente das demais, por isso AssetBundles podem ser carregados e descarregados sem afetar outra seção.

No entanto, texturas rosas aparecem às vezes.

Qual pode ser a causa desse problema?

A AssetBundle.mainAsset está quebrado.

B AssetBundle.Unload() está sendo chamado muito cedo.

C AssetBundle.LoadAllAssets() está sendo chamado com o Tipo errado.

D AssetBundle.LoadAssetWithSubAssets() está sendo chamado com Tipo e string errados.

Questão 5

O GDD é um jogo de quebra-cabeça de combinação 3D para dispositivos móveis com mais de 300 níveis. O jogo contém os seguintes eventos personalizados de Analytics para cada nível:

1. “levelStarted”: Desencadeado quando um jogador começa o nível
2. “levelCompleted”: Desencadeado quando um jogador conclui o nível

A equipe quer que você determine quais são os cinco níveis mais difíceis do jogo. Os níveis têm limite de tempo.

Qual evento personalizado adicional seria necessário para determinar os cinco mais difíceis?

A Adicionar um “levelTime” para enviar o tempo gasto no nível.

B Adicionar um “levelRestarted” para ser desencadeado quando o aplicativo móvel é fechado pelo sistema operacional.

C Adicionar um “levelFailed” para ser desencadeado quando um jogador falha ou sai do nível mais cedo.

D Adicionar um “levelResumed” para ser desencadeado quando o aplicativo móvel for retomado dos processos em segundo plano.

Respostas corretas: C, D, C, B, C