unity

unity

Certified

Expert

Gameplay

Programmer

Objetivos do Exame

Especialista Certificado Unity
Programador de Jogabilidade

Função

Os Programadores de Jogabilidade se concentram na execução do design do jogo. Esses Programadores dão vida ao jogo adicionando scripts a assets criados pela equipe artística. Eles são responsáveis por implementar a interface de usuário (UI), ambiente, personagens e objetos, além dos scripts e otimização necessários para executar a mecânica do jogo. Eles também dão funcionalidade a personagens não jogáveis (NPCs) que povoam o jogo. Programadores de Jogabilidade trabalham junto a designers (ou eles mesmos podem ser designers) e recorrem ao Game Design Document (GDD) para transformar o jogo em realidade. São pessoas altamente técnicas e entendem o que é preciso para dar vida às ideias. Isso inclui saber quando e como usar componentes Unity, criar seus próprios componentes, configurar Prefabs e organizar cada cena adequadamente. Isso tudo é feito com o Technical Design Document (TDD) em mente.

Títulos de trabalho para esta função

- Programador de Jogabilidade
- Engenheiro/Scripter de Jogo
- Designer de Níveis

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
- Experiência em projetar jogos imersivos, jogos web/móveis e jogos digitais
- Bom conhecimento de scripting e programação usando C#
- Experiência com o ciclo completo de produção de um jogo, desde o conceito inicial até o lançamento
- Compreensão completa de todos os aspectos do design de um jogo (níveis, personagens não jogáveis [NPCs], mecânicas de jogo)
- Experiência em projetar e implementar os Serviços do Unity
- Compreensão das pipelines de animação e assets de um jogo, incluindo configuração de personagens e ambientes
- Grande capacidade de organização no que diz respeito a estruturas de arquivos, convenções de nomenclatura e outros protocolos estabelecidos
- Habilidade de criar e usar uma biblioteca de Prefabs para fazer o melhor jogo possível

Habilidades essenciais

A certificação de Programador de Jogabilidade Especialista verifica que os candidatos tenham as habilidades necessárias para criar e otimizar um jogo de forma eficiente. Os candidatos bem sucedidos têm conhecimento avançado nas seguintes áreas.

Prototipagem

- Criar e avaliar protótipos de jogabilidade essencial para iteração rápida
- Projetar e implementar GameObjects para testar a funcionalidade de mecânicas do jogo
- Criar uma biblioteca de Prefabs com base no GDD, usando assets gerados pela equipe artística
- Organizar e configurar a estrutura de pastas para uma biblioteca de Prefabs

Design de Níveis

- Projetar e criar prefabs de nível interativo usando componentes Colliders e Rigidbody
- Definir a camada adequada para physics masking em prefabs interativos
- Configurar prefabs gerados em tempo de execução e implementar uma jogabilidade dinâmica
- Configurar e implementar desencadeadores de eventos e eventos com script usando Colliders como gatilhos
- Criar e fazer o script de componentes de lógica personalizados para GameObjects a fim de conectar máquinas de estados a interações/ gatilhos na cena
- Posicionar e configurar efeitos na cena, tanto estáticos quanto gerados em tempo de execução
- Configurar level of details (LOD) de acordo com as especificações da plataforma
- Avaliar o posicionamento de GameObject e dependências de acordo com as especificações da plataforma
- Avaliar a cena para streaming ou carregamento estático de cena
- Construir níveis do jogo com cenas múltiplas
- Configurar a cinemática para melhorar a jogabilidade

Design de Personagem não Jogável (NPC)

- Projetar e criar scripts de lógica e inteligência artificial (IA) para NPCs
- Implementar navegação e pathfinding para NPCs usando NavMeshes, NavMeshAgents, NavMesh obstacles e Off-Mesh links
- Configurar tipos e custos de área NavMesh
- Configurar gatilhos para ativar/desativar áreas NavMesh
- Implementar evasão NavMeshAgent e simulador de multidões
- Avaliar o posicionamento de NPCs e dependências de acordo com as especificações da plataforma
- Configurar áudio e efeitos frame-based em clipes de animação

Design de Interface de Usuário e Mecânica do Jogo

- Usar o Sistema de Animação para animator controllers, máquinas de estados e scripts de lógica para a mecânica do jogo
- Avaliar e otimizar componentes collider, Rigidbody e physics materials para GameObjects interativos
- Implementar interfaces de usuário relativas a jogabilidade, tais como HUDs, minimapas, sistemas de radar, barras de saúde e outros elementos de dados

Otimização de Desempenho e Plataformas Alvo

- Implementar o download e posicionamento de AssetBundles em níveis do jogo
- Projetar e modificar esquemas de entrada e controle para diferentes plataformas e/ou realidade virtual (RV)
- Analisar GameObjects e cenas para otimizar tempo de execução e armazenamento de acordo com as especificações da plataforma
- Otimizar occlusion culling nas cenas
- Depurar e testar níveis do jogo durante o tempo de execução

Implementação de Serviços do Unity: Ads, In-App Purchases e Analytics

- Implementar Unity Ads simples e recompensadores
- Implementar Unity In-App Purchasing (IAP)
- Projetar a integração de Unity Analytics no GDD e no TDD
- Configurar o monitoramento de eventos com dados personalizados usando o Analytics para monitorar o comportamento do jogador
- Analisar e avaliar níveis existentes e recomendar mudanças com base nos dados do Analytics
- Usar Unity Performance Reporting para modificar e otimizar builds do jogo de acordo com a plataforma

Tópicos do Exame de Certificação

Prototipagem (Jogabilidade Essencial para Iteração Rápida)

- Prototipagem da jogabilidade essencial
- Conflitos e soluções durante a fase de prototipagem

Programação de Design de Níveis

- Configuração de física
- Raycasting
- Prefabs gerados por scripts durante o tempo de execução
- Lógica e comportamento de níveis
- Povoamento de níveis com Particle Systems e efeitos
- Otimizações de plataforma
- Carregamento e descarregamento de cena
- Métodos de exibição de cinemática

Programação do Design de NPC

- Lógica e comportamento de NPC
- Navegação e pathfinding
- Geração e posicionamento de NPC

Implementação de Interface de Usuário

- Sistemas de coordenadas de UI e scripting de UI

Otimização de Desempenho e Plataformas Alvo

- Otimização de renderização
- Download e configuração de Asset Bundle
- Depuração de jogabilidade
- Diferenças entre plataformas e seu impacto na jogabilidade

Implementação de Serviços do Unity

- Unity Ads
- Unity In-App Purchasing
- Unity Analytics
- Unity Cloud Build

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: D, D, C, B, C