unity

unity

Certified

Programmer

Objetivos do Exame

Programador
Certificado Unity

Função

Profissionais de programação Unity desenvolvem conteúdo interativo usando Unity. Junto com outros membros da equipe de desenvolvimento (por exemplo, profissionais de áudio e arte), o programador Unity dá vida às ideias para o aplicativo usando os recursos do Unity Editor em conjunto com os assets visuais e de áudio criados pelos outros membros da equipe de desenvolvimento de software. O programador Unity é generalista, é capaz de resolver difíceis problemas de código e é responsável por contribuir em uma ampla gama de tarefas técnicas. Isso inclui integrar assets de arte, criar a interface do usuário, fazer scripts de interações do usuário e regras do sistema do jogo, implementar lógica de estado, simular física, depurar o código e otimizar o desempenho.

A certificação de Programador Certificado Unity tem como público-alvo programadores de nível iniciante a médio e alunos de curso superior que buscam funções de programação em uma variedade de setores. Esta certificação mostra a empregadores em potencial que seu titular:

- É perspicaz ao programar no contexto do processo de desenvolvimento profissional de softwares, sendo capaz de criar e manter aplicativos feitos com a engine Unity
- Tem destreza com processos técnicos, pensamento lógico e engenhosidade
- É capaz de ficar a cargo de tarefas de programação rotineiras de nível intermediário de forma independente e de trabalhar em desafios técnicos complexos ao lado de engenheiros mais experientes

Títulos de trabalho para esta função

- Programador de jogabilidade
- Engenheiro de software
- Desenvolvedor de software
- Desenvolvedor Unity
- Desenvolvedor de aplicativos móveis

Pré-requisitos

Essa certificação foi criada para programadores que acabaram de se formar em programação de jogos, ciência da computação ou campos relacionados, estudantes independentes com dois ou mais anos de estudo equivalente à faculdade ou experiência em programação, ou profissionais no início ou no meio da carreira que usam Unity no ambiente de trabalho. Os candidatos devem comparecer ao exame com experiência prática prévia em programação de aplicativos interativos com Unity por si só ou como parte de uma equipe multifuncional que tenha um protótipo ou demonstração técnica completos.

Pré-requisitos de experiência:

- Experiência de dois anos ou mais com programação interativa em 3D de jogos usando Unity
- Experiência de dois anos ou mais com programação, incluindo C#
- Experiência com o ciclo completo de desenvolvimento de software, trabalhando desde o conceito inicial até a conclusão
- Entendimento dos usos profissionais de desenvolvimento de software com Unity, incluindo desenvolvimento de jogos, entretenimento interativo e visualização de design
- Compreensão básica das pipelines de animação e assets visuais/3D em Unity, incluindo configuração de personagens e ambientes
- Compreensão das práticas de desenvolvimento de software profissional em equipe, incluindo teste e controle de versão
- Conhecimento dos Serviços Unity para colaboração, monetização, operações ao vivo e multijogador
- Compreensão matemática necessária para desenvolvimento interativo em 3D, incluindo álgebra linear e operações com matrizes

Observação: Essa certificação foi desenvolvida para a versão 2017.3 do Unity.

Habilidades essenciais

As habilidades essenciais nesse campo de trabalho se concentram na contribuição para a execução técnica de um projeto, do conceito ao lançamento e além.

Programação de interações essenciais

- Implementar e configurar física e comportamentos para GameObjects
- Implementar e configurar entradas e controles
- Implementar e configurar visualizações de câmera e movimento

Trabalhar na pipeline de arte

- Entender materiais, texturas e shaders e escrever scripts que interajam com a API de renderização do Unity
- Entender iluminação e escrever scripts que interajam com a API de iluminação do Unity
- Entender animação 2D e 3D e escrever scripts que interajam com a API de animação do Unity
- Entender o sistema de partículas e efeitos e escrever scripts que interajam com a API de sistema de partículas do Unity

Desenvolvimento de aplicativos

- Interpretar scripts de fluxo de interface para o aplicativo, tais como sistemas de menu, navegação de UI e configurações do aplicativo
- Interpretar scripts de personalização controlada pelo usuário, tais como criadores de personagens, inventários, fachadas e compras no aplicativo
- Analisar scripts de funcionalidades de progressão do usuário, tais como pontuação, níveis e economia dentro do jogo, usando tecnologias como Unity Analytics e PlayerPrefs
- Analisar scripts para overlays 2D como heads-up displays (HUDs), minimapas e anúncios
- Identificar scripts para salvar e recuperar dados do usuário e do aplicativo
- Reconhecer a avaliar o impacto da funcionalidade de multijogador e rede

Programação para design de ambiente e cena

- Determinar scripts para implementar assets de áudio
- Identificar métodos para implementar instanciação, destruição e gerenciamento de GameObjects
- Determinar scripts de pathfinding com o sistema de navegação do Unity

Otimização para desempenho e plataformas

- Avaliar erros e problemas de desempenho usando ferramentas como o Unity Profiler
- Identificar otimizações e lidar com requisitos para configurações de hardware e/ou plataformas específicas
- Determinar otimizações e affordances de UI comuns para as plataformas RX

Trabalhar em equipes de desenvolvimento de software profissionais

- Reconhecer conceitos associados aos usos e impactos do controle de versão, usando tecnologias como Unity Collaborate
- Demonstrar conhecimento de testes de desenvolvedor e seu impacto no processo de desenvolvimento de software, incluindo Unity Profiler e técnicas de teste e depuração tradicionais
- Reconhecer técnicas de estruturação de scripts para modularização, leitura e reutilização

Tópicos do Exame de Certificação

Programação de interações essenciais

- Implementar comportamentos e interações de GameObjects e ambientes
- Identificar métodos para implementar entradas e controles
- Identificar métodos para implementar visualizações de câmera e movimento

Trabalhar na pipeline de arte

- Conhecimento de materiais, texturas e shaders—API de renderização do Unity
- Conhecimento de iluminação—API de iluminação do Unity
- Conhecimento de animação 2D e 3D—API de animação do Unity
- Conhecimento de sistemas de partículas—API de partículas do Unity

Desenvolvimento de aplicativos

- Fluxo de interface para o aplicativo, tais como sistemas de menu, navegação de UI e configurações do aplicativo
- Personalização controlada pelo usuário, tais como criadores de personagens, inventários, fachadas e compras no aplicativo
- Implementar funcionalidades de progressão do usuário, tais como pontuação, níveis e economia dentro do jogo, usando ferramentas como Unity Analytics
- Implementar overlays 2D como heads-up displays (HUDs), minimapas e anúncios
- Salvar e recuperar dados do usuário e do aplicativo
- Reconhecer o valor e o impacto da funcionalidade de multijogador e rede

Programação para design de ambiente e cena

- Determinar scripts para implementar assets de áudio
- Identificar métodos para implementar instanciação, destruição e gerenciamento de GameObjects
- Determinar scripts de pathfinding com o sistema de navegação do Unity

Otimização para desempenho e plataformas

- Avaliar erros e problemas de desempenho usando ferramentas como o Unity Profiler
- Identificar otimizações e lidar com requisitos para configurações de hardware e/ou plataformas específicas
- Determinar otimizações e melhorias de UI comuns para as plataformas RX

Trabalho em equipes de desenvolvimento de software

- Controle de versão: Impactos e usos de ferramentas como Unity Collaborate
- Testes e seu impacto no processo de desenvolvimento de software
- Reconhecer técnicas de estruturação de scripts para modularização, leitura e reutilização

Exemplos de questões

Questão 1

Um programador deve implementar um sistema de menu de UI. Cada menu consiste de um painel de UI e um ou mais botões de UI, todos filhos de um objeto UI Canvas. Todo o sistema de menu da UI será criado em uma cena separada que é carregada de forma adicional.

O estilo da arte dos painéis e botões deve ser consistente (cor, textura, tipo de transição do botão, etc.), mas a diretora de arte ainda não tomou essas decisões. Ela quer trabalhar nessas configurações junto com o trabalho do programador na UI. As mudanças dela seriam efetuadas em todos os objetos da cena, novos e já existentes.

Qual seria a melhor maneira do programador usar os recursos do Unity para criar facilmente um sistema de menu funcional e que permita que a diretora de arte trabalhe ao mesmo tempo (e de forma independente) na aparência?

A Criar subclasses para `UI.Button` e `UI.Panel` e definir os valores da aparência em código.

B Criar novos materiais de botão e painel e atribui-los a todos os botões e painéis na cena.

C Usar prefabs para o botão e o painel e falar para a diretora de arte modificar os prefabs.

D Escrever um script para buscar/substituir valores no arquivo da cena de acordo com as decisões da diretora de arte.

Questão 2

Um jogo de corrida infinita em 3D se passa em vários trilhos de trem paralelos em um pátio ferroviário. O jogador sempre corre para a frente nos trilhos e precisa evitar trens vindo na sua direção pulando sobre eles ou para o trilho adjacente.
Cada trem novo adicionado ao trilho é adicionado atrás dos outros trens naquele trilho.
No entanto, os trens às vezes se sobrepõem porque se movem na direção do jogador em diferentes velocidades, ou não se movem, e isso precisa ser corrigido.

Qual é a melhor forma, em termos de desempenho, de prevenir que novos trens se sobreponham aos trens que já estão naquele mesmo trilho?

A Ao gerar um trem no trilho, determinar uma posição de geração que evite o problema usando as velocidades do novo trem e do último trem colocado naquele trilho, além do ponto em que o último trem colocado no trilho desaparecerá ao passar pelo jogador.

B Quando um trem estiver se movendo, aplicar raycast à frente da parte da frente do trem e empurrar qualquer trem atingido pelo raycast para a frente com a velocidade do trem mais rápido.

C Ao gerar um trem no trilho, adicionar um Rigidbody a ele e então usar forças para mover os trens.

D Ao gerar um trem no trilho, usar BoxCast com um comprimento proporcional à velocidade do trem para garantir que ele não colida com outro trem até estar atrás da câmera.

Questão 3

Um programador está trabalhando em uma sala escura e melancólica e precisa criar uma tocha tremulante que lança uma sombra misteriosa nas paredes, chão e teto. O programador escreve essas funções em um `MonoBehaviour` anexado à tocha:

```
void Start()
{
    Light light = GetComponent<Light>();
    light.lightMapBakeType = LightMapBakeType.Mixed;
    light.type = LightType.Area;
    light.shadows = LightShadows.Soft;
    light.range = 5f;
}
void Update()
{
    GetComponent<Light>().intensity = Mathf.PerlinNoise(Time.time, 0);
}
```

A tocha não lança nenhuma luz nem sombra no tempo de execução. A luz está configurada com os valores padrão no Unity Editor.

O que o programador deve mudar para esse código funcionar como necessário?

A Definir `light.lightBakeType` como `LightmapBakeType.Realtime`

B Definir `light.range` como `10`

C Definir `light.shadows` como `LightShadows.Hard`

D Definir `light.type` como `LightType.Point`

Questão 4

Um programador está desenvolvendo um jogo de simulação de mineração em que o jogador pode cavar o chão em busca de minerais. Em um dos locais, o jogador pode criar um túnel que atravessa um sistema de cavernas já existente. A documentação do design especifica que qualquer áudio que aconteça tanto nas cavernas atuais quanto nos novos túneis deve ter reverberação. O programador precisa garantir que o usuário esteja de forma consistente na ReverbZone da caverna mais próxima.

Como o programador deve manipular as propriedades de AudioReverbZone para cumprir esses requisitos?

A Aumentar as reflexões para combinar com a nova área.

B Aumentar maxDistance das duas ReverbZones para que se toquem na nova área de conexão.

C Aumentar a reverberação para acomodar a nova área.

D Aumentar decayTime na nova área.

Questão 5

Ao escrever uma função de carregamento, um programador recebe um erro de compilação:

```
error CS1624: The body of `CustomAnalytics.LevelLoading()' cannot be
an iterator block because `void' is not an iterator interface type
```

```
void LevelLoading(){
    AsyncOperation async = SceneManager.LoadSceneAsync("Level_01");
    while (!async.isDone)
    {
        yield return null;
    }
}
```

O que o programador deve fazer para corrigir esse erro?

A Alterar `yield return null` para `yield return WaitForSeconds(0)`

B Alterar `void LevelLoading()` para `IEnumerator LevelLoading`

C Alterar `SceneManager.LoadSceneAsync("Level_01")` para `Application.LoadLevelAdditiveAsync("Level_01")`

D Alterar `while (!async.isDone)` para `while (!async.allowSceneActivation)`

Questão 6

O sistema de entrada de um jogo de corrida é mapeado de forma que o eixo de entrada horizontal controla a direção. Durante os testes, descobriram que alguns dispositivos de controle registram entrada de direção mesmo com o direcional centralizado.

Qual mudança deve ser feita no eixo no sistema de entrada para resolver esse problema?

A Aumentar Gravity

B Definir Snap como true

C Aumentar Deadzone

D Diminuir Sensitivity

Questão 7

Em um jogo de aventura que se passa em um planeta alienígena, o jogador deve exterminar várias formas de vida. A pontuação do jogador aumenta com cada morte. A documentação do design diz que a pontuação deve estar ligada à conta do jogador para poder ser recuperada mais tarde, mesmo se o jogador estiver em outra sessão de jogo ou usando um dispositivo diferente.

Qual é o método mais confiável para o programador armazenar os dados da pontuação?

A Usar `DontDestroyOnLoad()` no GameObject com os dados da pontuação e fazer upload dos dados em um servidor antes do aplicativo fechar.

B Salvar a pontuação em `PlayerPrefs` toda vez que for atualizada e enviada para um servidor antes do aplicativo fechar.

C Usar um valor estático para armazenar os dados da pontuação para que eles estejam disponíveis na próxima sessão de jogo .

D Usar serialização de dados para armazenar a pontuação de forma persistente e fazer upload para um servidor.

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