unity

試験範囲

Unity 認定
3D アーティスト

役割

Unity 3D アーティストの主な役割は、Unity Engine を使用してリアルタイムでレンダリングされるインタラクティブソフトウェアに 3D アートを実装することです。3D アーティストはビジュアルアセットを Unity に組み込み、ゲームまたはアプリケーションの「世界」を構築します。また、レイヤーでビジュアル情報の追加や操作を行って、創造性に満ちた映像をプロジェクトで実現するうえで重要な役割を果たします。3D アーティストは、独特のスタイルやムードでオブジェクト、キャラクター、環境を演出することに長けたジェネラリストです。3D モデルへのマテリアルとシェーダーの適用、3D 環境の設定と管理、カメラの設定と制御、シーンのライティング、パーティクルエフェクトの使用など、アプリケーションのエステティックに関するタスクで幅広く活躍します。

Unity 認定 3D アーティストは、さまざまな業界で職を探している、エントリーレベルから中級レベルのアーティストおよび大学や専門学校の卒業予定者を対象としたプロフェッショナル認定資格です。この認定資格があれば、採用担当者に以下を示すことができます。

- プロフェッショナルなソフトウェア開発プロセスにおいて、デザインアセットからゲーム世界を作り上げる芸術的スキルと技術的スキルを兼ね備えていること。
- 創造力と表現力を備えているうえ、自身の仕事の技術的な面にも明るく、プログラミング用語の基礎的な知識を用いて、より高度な技術的知識を持ったチームメンバーとコミュニケーションできること。
- リアルタイム 3D アプリケーションに求められる「ルックアンドフィール」を実現できること。また、ユーザーインターフェースとオブジェクトの動きのプロトタイプ作成に役立つ基本的な 2D アートとアニメーションを活用できること。

この役割に対応する職種

- 3D アーティスト
- 3D ジェネラリスト
- ゲームアーティスト
- レベルデザイナー
- 環境アーティスト
- 3D ビジュアライゼーションアーティスト

前提条件

ゲーム、デザインビジュアライゼーション、または Unity Engine で実現可能な各種アプリケーションの制作に携わるリアルタイム 3D アーティストとして職業キャリアを開始する準備ができたアーティスト。このようなアーティストに該当するのは、ゲームアート、コンピューターグラフィックス、またはその関連分野の大学を最近卒業した者、2 年以上の大学相当の教育を修了しているか 3D モデリングの実務経験を持つ自立した学習者、あるいは、すでに初心者として仕事を始めているプロフェッショナルです。バックグラウンドに関わらず、個人またはクロスファンクショナルチームの一員として Unity で 3D オブジェクトおよび環境を実際に実装した経験を持ち、プロトタイプまたはデモリールを完成させている必要があります。これらのアーティストは、Unity に関するスキルのテストと評価のほか、求人市場での魅力を高めることを目的としてこの認定試験を受験します。

前提条件の内容：

- Unity を使用して作成されたビデオゲームやその他のリアルタイム 3D アプリケーションへの 3D アートおよび 3D 環境の実装の実務経験
- PC、モバイル端末、XR などの各種プラットフォーム向けに Unity で 3D オブジェクトと 3D 環境のインポート、設定、ライティングを行った経験
- 初期構想から完成までソフトウェア開発ライフサイクル全般に携わった経験
- 3D 環境とアプリケーション UI のプロトタイプ作成の経験
- アニメーションと 2D レンダリングに対する基本的な知識
- Unity のプログラミングワークフローと用語に対する基本的な知識
- デザインドキュメンテーションとバージョン管理など、プロフェッショナルなソフトウェア開発手法に対する理解
- ゲーム開発の経験、またはシミュレーションやデザインビジュアライゼーションなどのリアルタイム 3D アプリケーション開発の経験

注：この認定資格は、Unity バージョン 2017.3 を元に作成されました。

中心的なスキル

3D オブジェクトのレンダリング

- 3D アセットを Unity にインポートするために適切なインポート設定を選択する
- インポートした 3D アセットに関する一般的な問題のトラブルシューティングを行う
- マテリアルをオブジェクトに追加し、高度なマテリアル設定を使用して目的のエフェクトを実現する
- テクスチャをマテリアルに追加し、高度なテクスチャ設定を使用して目的のエフェクトを実現する
- Unity のスタンダードシェーダーの設定を操作して目的のエフェクトを実現する
- カメラのプロパティーを調整して目的のエフェクトを実現する
- Level of Detail（LOD）グループとオブジェクトを使用してシーンを最適化する

シーンと環境デザインのプログラミング

- オーディオアセットを実装するためのスクリプトを決定する
- ゲームオブジェクトのインスタンス化、破壊、管理を実装する方法を特定する
- Unity ナビゲーションシステムによる経路探索のスクリプトを決定する

パフォーマンスとプラットフォームに合わせた最適化

- Unity Profiler などのツールを使用してエラーやパフォーマンスの問題を評価する
- 特定のビルドプラットフォームやハードウェア構成の要件に対処するための最適化を確認する
- XR プラットフォーム向けの一般的な UI アフォーダンスおよび最適化を決定する

プロフェッショナルなソフトウェア開発チームへの参加

- Unity Collaborate などの技術を使用して、バージョン管理の使用および影響に関連した概念を理解する
- Unity Profiler、従来のデバッグ手法やテスト手法など、ソフトウェア開発プロセスにおける開発者テストとその影響に関する知識を明らかにする
- モジュール性、可読性、再利用性を実現するためのスクリプトを構成する手法を理解する

認定試験のトピック

中核となるインタラクションのプログラミング

- ゲームオブジェクトおよび環境の動作とインタラクションを実装する
- 入力とコントロールを実装する方法を特定する
- カメラビューとカメラ動作を実装する方法を特定する

アートパイプラインでの操作

- マテリアル、テクスチャ、シェーダーの知識 – Unity レンダリング API
- ライティングの知識 – Unity ライティング API
- 2D アニメーションおよび 3D アニメーションの知識 – Unity アニメーション API
- パーティクルシステムの知識 – Unity パーティクル API

アプリケーションシステムの開発

- メニューシステム、UI ナビゲーション、アプリケーション設定などのアプリケーションインターフェースフロー
- キャラクター作成システム、インベントリ、ストアフロント、アプリ内課金（IAP）など、ユーザーが制御するカスタマイズ
- Unity Analytics などのツールを使用したスコア、プレイヤーレベル、ゲーム内経済など、ユーザーの進行度に関する機能を実装する
- ヘッドアップディスプレイ（HUD）、ミニマップ、広告などの 2D オーバーレイを実装する
- アプリケーションデータとユーザーデータの保存と取得
- ネットワーキング機能とマルチプレイヤー機能の価値と影響を認識する

シーンと環境デザインのプログラミング

- オーディオアセットを実装するためのスクリプトを決定する
- ゲームオブジェクトのインスタンス化、破壊、管理を実装する方法を特定する
- Unity ナビゲーションシステムによる経路探索のスクリプトを決定する

パフォーマンスとプラットフォームに合わせた最適化

- Unity Profiler などのツールを使用してエラーやパフォーマンスの問題を評価する
- 特定のビルドプラットフォームやハードウェア構成の要件に対処するための最適化を確認する
- XR プラットフォーム向けの一般的な UI アフォーダンスおよび最適化を決定する

ソフトウェア開発チームへの参加

- バージョン管理：Unity Collaborate などのツールの影響と使用
- テストおよびソフトウェア開発プロセスに対するその影響
- モジュール性、可読性、再利用性を実現するためのスクリプトを構成する手法の理解

サンプル問題

問題 1

あるアーティストが自動車のビジュアライゼーションを制作しています。モデル自体は非常に細かいメッシュで、10 万個の頂点があります。ビジュアライゼーションのためには、正確なレンダリングとメッシュの詳細が必要です。モデルをシーンビューウィンドウに取り込むと、いくつかのサブメッシュに分かれて表示されてしまいます。これではレンダリングが不自然になり、平滑化もうまくできません。

この問題を修正するためには、インポート設定をどのように変更する必要がありますか？

A `UI.Button` と `UI.Panel` のサブクラスを作成し、ルックアンドフィールの値をプログラムで設定する。

B 新しいボタンとパネルのマテリアルを作成し、シーンのすべてのボタンとパネルに割り当てる。

C ボタンとパネルにプレハブを使用し、アートディレクターにプレハブを修正してもらう。

D アートディレクターの入力内容に基づいてシーンファイル内の値を検索 / 置換するスクリプトを作成する。

問題 2

車両基地の中に複数の線路が並行して走っている、3D のエンドレスランナーゲームがあります。プレイヤーは線路上を常に前に向かって走っており、対向列車が来た場合は、飛び越えたり隣の線路に飛び移ったりして避ける必要があります。
新しい列車は、線路に追加されるたびに、その線路上のその他すべての列車の後ろに追加されます。ところが、対向列車のスピードはさまざまであるため、たまに列車どうしが重なることがあります。これは、修正する必要があります。

新しい列車が同じ線路上の既存の電車と重ならないようにするのに最も効率的な方法はどれですか？

A 線路上に列車を生成するときに、新しい列車と線路に最後に配置した列車の速度、および線路に最後に配置した列車がプレイヤーを通過したときに消滅する位置を使用して、この問題を回避する生成位置を決定する。

B 動いている列車の前面から前方に向かってレイキャストし、レイキャストでヒットした列車をより速い列車速度を使用して前に押し出す。

C 線路上に列車を生成する場合、その列車に Rigidbody を追加し、フォースを使って列車を動かす。

D 線路上に列車を生成する場合、長さが列車の速度に比例した BoxCast を使用して、列車がカメラを通過するまで他の列車と衝突しないようにする。

問題 3

プログラマーは暗くて雰囲気のある部屋を作成しており、壁、床、天井に不気味な影を映し出すちらちらと揺らめくたいまつを作成する必要があります。プログラマーは、たいまつにアタッチされた MonoBehaviour について、次の関数を記述しています。

```
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);
}
```

実行時、たいまつは光も影も作りません。Unity Editor で、ライトはデフォルト値に設定されています。

このコードを仕様どおりに動作させるには、プログラマーは何を変更する必要がありますか？

A `light.lightBakeType` を `LightmapBakeType.Realtime` に設定する

B `light.range` を `10` に設定する

C `light.shadows` を `LightShadows.Hard` に設定する

D `light.type` を `LightType.Point` に設定する

問題 4

プログラマーは採掘シミュレーションゲームを開発しています。このゲームでは、プレイヤーが鉱物を探し求めて地面を掘り進めることができます。サイトの 1 つで、プレイヤーは既存の洞窟と交差するトンネルを作成することができます。デザイン ドキュメントでは、現在の洞窟と新しいトンネルの両方で再生されるオーディオに残響が生じるよう指定されています。プログラマーは、ユーザーが最も近い洞窟の ReverbZone に常に属しているようにする必要があります。

上記の要件を満たすために、プログラマーは AudioReverbZone のプロパティーをどのように操作する必要がありますか？

A 新しい領域に合わせてリフレクションを大きくする。

B 両方の ReverbZone の maxDistance を大きくして、これらが新しい連結領域内で接触するようにする。

C 新しい領域に合わせて残響を大きくする。

D 新しい領域の decayTime を大きくする。

問題 5

プログラマーが読み込み関数の記述中に次のコンパイルエラーが発生しました。

```
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;
    }
}
```

このエラーを修正するためにプログラマーはどうすればよいですか？

A `yield return null` を `yield return WaitForSeconds(0)` に変更する

B `void LevelLoading()` を `IEnumerator LevelLoading` に変更する

C `SceneManager.LoadSceneAsync("Level_01")` を `Application.LoadLevelAdditiveAsync("Level_01")` に変更する

D `while (!async.isDone)` を `while (!async.allowSceneActivation)` に変更する

問題 6

ドライビングゲームの入力システムは、水平入力軸でステアリングを制御するようにマッピングされています。テスト中、一部のジョイスティックデバイスで、スティックが中央にあってもステアリング入力が登録されることがわかりました。

この問題を解決するために、入力システムの軸にどのような変更を加える必要がありますか？

A「Gravity」を大きくする。

B「Snap」を true に設定する。

C「Deadzone」を大きくする。

D「Sensitivity」を小さくする。

問題 7

エイリアンのいる惑星を舞台としたアドベンチャーゲームで、プレイヤーはさまざまな生命体を倒す必要があります。プレイヤーが生命体を倒すたびにスコアが加算されます。デザインドキュメントには、プレイヤーが別のセッションや別のデバイスでプレイした場合でも後でスコアを取り戻せるようにするために、スコアをプレイヤーのアカウントにリンクする必要があると記述されています。

プログラマーがスコアデータを保存するための最も信頼性の高い方法はどれですか？

A スコアデータを保持するゲームオブジェクトで `DontDestroyOnLoad()` を使用して、アプリケーションの終了直前にサーバーにデータをアップロードする。

B スコアが更新され、アプリケーションの終了直前にサーバーにアップロードされるたびに `PlayerPrefs` にそのスコアを保存する。

C 静的な値を使用してスコアデータを保存し、次のプレイセッションで使用できるようにする。

D データのシリアル化を使用してスコアデータを永続的に保存し、サーバーにアップロードする。

正解：C、A、D、B、B、C、D