unity

unity

Certified

Expert

Gameplay

Programmer

測驗目標

Unity 專家級認證
遊戲程式設計師

角色

遊戲程式設計專家著眼於遊戲設計的執行面。遊戲程式設計師爲美術建立的資源設計腳本，爲遊戲注入生命。他們負責實作使用者介面 (UI)、環境、角色和物件，以及執行遊戲機制所需的最佳化和腳本。他們還爲遊戲中的非玩家角色 (NPC) 賦予功能。

遊戲程式設計師與設計師合作 (或者他們本身就是設計師)，參考遊戲設計文件 (GDD)，將遊戲化爲現實。這群人技術專精，熟知如何將概念變成現實。這包括熟悉 Unity 元件的使用時機與使用方法、能夠編寫自己的元件、設定 Prefab，以及適當地組織每個場景。他們在技術設計文件 (TDD) 的基礎上完成所有這一切。

此角色的職務名稱

- 遊戲程式設計師
- 遊戲腳本設計師/工程師
- 關卡設計師

先決條件

本專家級認證推薦給那些在這個領域已有多年經驗，且累積許多先進與實際應用經驗的人，例如：

- 有在遊戲公司開發的經驗
- 設計沉浸式遊戲、網頁/手遊和數位遊戲的經驗
- 熟知如何以 C# 編寫腳本/程式碼
- 曾親身參與從早期概念到遊戲上市的完整遊戲開發過程
- 精通遊戲設計的所有領域 (關卡、非玩家角色 [NPC]、遊戲機制)
- 有設計與實作 Unity Services 的經驗
- 了解遊戲資源與動畫管線，包括角色與環境設定
- 有扎實的組織能力，能遵守檔案結構、命名慣例和其他已建立的協定
- 能夠建立和使用 Prefab 庫，以打造最佳的遊戲

核心技能

遊戲程式設計師專家認證是要確認候選人具備所需的技能，能夠有效組合並將遊戲最佳化。通過的考生代表精通下列領域。

雛形製作

- 製作並評估核心遊戲雛形以加速開發
- 設計並實作 GameObjects 以驗證遊戲機制的功能
- 使用美術團隊產生的資源，依照 GDD 建立 Prefab 庫
- 爲 Prefab 庫組織並設定資料夾結構

關卡設計

- 使用 Collider 和 Rigidbody 元件設計和製作互動式 Prefab
- 在互動式 Prefab 上爲物理遮罩設定適當的 layer
- 設定執行時產生的 Prefab，並實現動態遊戲
- 使用 Collider 作爲觸發器來設定和實現事件觸發器和腳本事件
- 建立和編寫用於 GameObjects 的自訂邏輯元件，以便將狀態機連結到場景中的互動/觸發器
- 在整個場景中放置和設定效果，包括靜態和執行時產生的位置
- 根據平台規範設定多層次細節 (LOD)
- 根據平台規範評估 GameObject 的放置與相依性
- 評估串流場景與靜態場景載入
- 用多個場景建構遊戲關卡
- 設定動畫以加強遊戲體驗

非玩家角色 (NPC) 設計

- 設計並製作 NPC 邏輯和人工智慧 (AI) 腳本
- 使用 NavMesh、NavMeshAgent、NavMesh Obstacle 和 Off-Mesh Link 實現 NPC 導航和尋徑
- 設定 NavMesh 的區域類型與效能成本
- 設定觸發器以啓用/停用 NavMesh 區域
- 實作 NavMeshAgent 迴避和人群模擬
- 根據平台規範評估 NPC 的放置與相依性
- 在動畫片段中設定基於幀算法的音效與效果

使用者介面(UI)與遊戲機制設計

- 在動畫控制器、狀態機使用動畫系統，以及在遊戲機制使用邏輯腳本
- 評估並將 Collider、Rigidbody 元件及 Physic Material 最佳化以製作互動式 GameObjects
- 實作與遊戲進行相關的使用者介面，例如：抬頭顯示器、小地圖、雷達系統、健康狀況列和其他由資料驅動的元素

效能最佳化與目標平台

- 在遊戲關卡中實作 AssetBundle 的下載與安置
- 針對不同平台及/或虛擬實境 (VR) 設計並修改輸入及控制器配置
- 分析 GameObjects 和場景，根據平台規範進行執行與儲存的最佳化
- 在整個場景中將遮擋剔除最佳化
- 在執行時對遊戲關卡進行除錯與測試

Unity Services 實作：Ads、In-App Purchases 與 Analytics

- 實作簡單的獎勵式廣告 Unity Ads
- 實作 Unity 內購 In-App Purchasing (IAP)
- 在 GDD 與 TDD 中設計 Unity Analytics 整合
- 使用 Analytics 設定對自訂資料事件的監測以掌握玩家行爲
- 分析和評估現有的關卡，並根據 Analytics 資料提出變更建議
- 使用 Unity Performance Reporting 依照平台對遊戲組建進行修改及最佳化

認證測驗主題

雛形製作 (加速迭代用的核心遊戲)

- 核心遊戲雛形製作
- 雛形製作階段的衝突與解決方案

關卡設計的程式設計

- 物理配置
- 射線投射 (Raycasting)
- 執行時由腳本產生的 Prefab
- 關卡邏輯與行爲
- 產生具有粒子系統與特效的關卡
- 平台最佳化
- 場景載入與卸載
- 顯示動畫的方法

NPC 設計的程式設計

- NPC 邏輯與行爲
- 導航與尋徑
- NPC 產生與放置

使用者介面實作

- UI 座標系統與 UI 腳本

效能最佳化與目標平台

- 渲染最佳化
- Asset Bundle 下載與設定
- 遊戲除錯
- 平台差異與對遊戲的影響

Unity Services 實作

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

範例問題

問題 1

一名程式設計師正在製作一個橫向捲軸射擊遊戲的雛形。主角是一架飛機，而敵人則是各種不同的幽浮。當玩家擊毀幽浮時，在幽浮原本的位置會出現爆炸效果。當玩家被擊落時，則會出現玩家機體爆炸的特效。

在螢幕上飛機可以發射多達 64 發子彈。螢幕上可顯示的任意類型幽浮最大數量是 128。螢幕上幽浮子彈的最大數量是 1024。螢幕上爆炸效果的最大數量是 128。

程式設計師需要確定哪些 GameObjects 是在編輯器中已經產生好，哪些應該在執行期間產生 (即透過物件池)。

問題：如何完成上述目標？

A 玩家、幽浮、玩家爆炸效果及幽浮爆炸效果應該在編輯器中產生。玩家的子彈和幽浮的子彈應該在 Runtime 執行時產生。

B 玩家、玩家子彈、幽浮及幽浮子彈應該在編輯器中產生。玩家爆炸效果及幽浮爆炸效果應該在 Runtime 執行時產生。

C 玩家及幽浮應該在編輯器中產生，而玩家子彈、幽浮子彈、玩家爆炸效果及幽浮爆炸效果在 Runtime 執行時產生。

D 玩家及玩家爆炸效果應該在編輯器中產生。幽浮、幽浮子彈、玩家子彈及幽浮爆炸效果應該在 Runtime 執行時產生。

問題 2

遊戲設計文件 (GDD) 定義了一個第三人稱開放世界遊戲。GDD 指定了兩種主角的遊戲方式：

1. 在世界中徒步行走
2. 騎摩托車四處移動

每個模式的相對高度是接近的，但是當玩家騎上摩托車時，他們的移動速度會比步行快得多。

程式設計師決定了在步行和騎車的切換過程中必須調整的幾個區域：

- 鏡頭視角 (FOV)
- 多層次細節 (LOD) 的距離
- 關卡串流 (Level streaming)

問題：當玩家騎上摩托車時，程式設計師應該用哪一種策略來依照 GDD 規範處理每個區域？

A 鏡頭 FOV 必須使用較小的值。
關卡必須以更快的速度串流。

B 鏡頭 FOV 必須使用較大的值。
關卡必須以更慢的速度串流。

C LOD 距離必須使用較大的值。
關卡必須以更慢的速度串流。

D 鏡頭 FOV 必須使用較大的值。
關卡必須以更快的速度串流。

問題 3

GDD 定義了一個大型城市，玩家可以在其中四處走動，並與城市中的任何市民互動。玩家可以幫市民解決各種隨機任務。玩家最多同時只能有 5 個進行中的任務。如果玩家從城市的一端走到另一端，大約需要 4 小時。

遊戲包含一個在抬頭顯示器 (HUD) 上的小地圖，其中顯示了下列項目的 2D 圖示：

- 玩家的位置
- 所有進行中的任務
- 在城市中走動的所有市民

小地圖可以放大和縮小。當它完全放大時，只會顯示玩家的圖示。當它完全縮小時，可以看見整個城市 25% 的區域。只有當玩家將小地圖完全放大時，才會出現延遲現象。

問題：造成遊戲變慢的原因可能是什麼？

A 從記憶體卸載所有市民圖示貼圖的呼叫太多。

B 任務圖示過於頻繁的查詢有多少任務正在進行。

C 小地圖的剔除算法花費太長的時間來剔除大部分城市。

D HUD 太過複雜，每一幀都需要重繪。

問題 4

GDD 定義了一款手遊，對於記憶體有很嚴格的要求。這個遊戲有用到 AssetBundle 來載入遊戲的不同部分，以確保遊戲進行順利。每個部分都各自獨立，因此可以在不影響其他部分的情況下載入和卸載 AssetBundle。

不過，有時候會出現粉紅色貼圖。

問題：可能導致問題的原因是以下哪一個？

A AssetBundle.mainAsset 已損毀。

B 太早呼叫 AssetBundle.Unload()。

C AssetBundle.LoadAllAssets() 被錯誤的型別呼叫。

D AssetBundle.LoadAssetWithSubAssets() 被錯誤的字串和型別呼叫。

問題 5

GDD 定義了一款 3D 的三消手遊，有超過 300 個關卡。遊戲的每個關卡包含了下列 Analytics 自訂事件：

1. “levelStarted”：當玩家開始關卡時觸發
2. “levelCompleted”：當玩家完成關卡時觸發

設計團隊想要確定遊戲中最難的 5 個關卡是哪些。關卡採取計時制。

問題：還需要哪一個自訂事件才能判斷最難的 5 個關卡？

A 加入一個 “levelTime” 事件來傳送過關所花的時間。

B 加入一個 “levelRestarted” 事件，當 App 被作業系統關閉時觸發。

C 加入一個 “levelFailed” 事件，當玩家任務失敗或提早退出關卡時觸發。

D 加入一個 “levelResumed” 事件，當 App 從背景處理序恢復時觸發。

正確答案：D、D、C、B、C