unity

시험 목표

Unity 인증
프로그래머

역할

Unity 프로그래밍 전문가는 Unity를 사용하여 인터랙티브 콘텐츠를 개발합니다. Unity 프로그래머는 오디오 및 아트 전문가와 같은 개발 팀의 다른 멤버와 함께, 소프트웨어 개발 팀의 다른 멤버가 생성한 비주얼/오디오 에셋과 Unity 에디터의 기능을 사용하여 애플리케이션의 비전을 실현합니다. Unity 프로그래머는 다양한 분야의 전문가입니다. 어려운 코딩 문제도 능숙하게 해결할 뿐 아니라, 아트 에셋 통합, 사용자 인터페이스 코딩, 사용자 인터랙션과 게임 시스템 규칙 스크립팅, 애플리케이션 상태 로직 구현, 물리 시뮬레이션, 코드 디버깅, 성능 최적화 등 광범위한 기술 작업에 기여할 책임이 있습니다.

Unity 인증 프로그래머는 초급부터 중급 레벨의 프로그래머와, 다양한 산업 분야에서 프로그래머가 되기를 지망하는 중고등교육 수료 학생을 위한 전문 인증 시험입니다. 본 인증 시험에 합격하면 향후 지원할 회사에 본인이 다음 자격을 갖추었다는 것을 증명할 수 있습니다.

- 전문적인 소프트웨어 개발 프로세스의 맥락에서 프로그래밍 감각을 발휘하여 Unity 엔진으로 빌드된 애플리케이션을 만들고 관리할 수 있음
- 기술 처리 과정에 있어서 재능이 있으며, 논리 지향적이면서 기지가 풍부함
- 일상적인 작업이나 중급 수준의 프로그래밍 작업은 스스로 독립적으로 처리하고, 복잡한 기술적 과제들은 선임 엔지니어와 함께 해결할 수 있음

해당 업무를 담당하는 직책

- 게임플레이 프로그래머
- 소프트웨어 엔지니어
- 소프트웨어 개발자
- Unity 개발자
- 모바일 애플리케이션 개발자

필수 조건

이 인증 시험은 최근에 대학을 졸업한 게임 프로그래밍, 컴퓨터공학 또는 관련 학과 전공자, 프로그래밍 분야에서 대학 교육에 준하는 과정을 수료했거나 업무 경력이 2년 이상 있는 개별 학습자, 업무와 관련하여 Unity를 사용하고 있는 초급에서 중급 수준의 경력을 가진 전문가를 위해 만들어졌습니다. 응시자는 단독으로 참여했거나 협업팀(Cross-Functional Team, CFT)의 일원으로서 참여하여 Unity로 인터랙티브 애플리케이션을 프로그래밍하여 완전한 프로토타입 또는 기술 데모를 완성해 본 실전 경험이 있어야 시험에 응시할 수 있습니다.

필수 조건:

- Unity를 사용한 비디오 게임 또는 3D 인터랙티브 프로그래밍 업무에서 실무 경험 2년 이상
- 컴퓨터 프로그래밍 업무에서 실무 경험 2년 이상(C# 포함)
- 개념 설계부터 완성까지 진행하는 전체 소프트웨어 개발 라이프사이클 경험
- Unity로 소프트웨어를 개발하기 위해 필요한 전문 애플리케이션(게임 개발, 인터랙티브 엔터테인먼트 및 디자인 시각화 등)에 대한 이해
- 캐릭터 및 환경 설정과 같은 Unity의 비주얼/3D 에셋 및 애니메이션 파이프라인에 대한 기본적 이해
- 유닛 테스트와 버전 관리와 같은 전문적인 소프트웨어 팀 개발 관행에 대한 이해
- 협업, 수익화, 실시간 운영 및 멀티플레이어를 위한 Unity 서비스에 대한 지식
- 선형대수학 및 행렬 연산과 같이 3D 인터랙티브 개발에 중요한 수학의 이해

참고: 이 인증 시험은 Unity 버전 2017.3용으로 개발되었습니다.

핵심 기술

이 업무 분야의 핵심 기술은 개념 설계부터 출시, 그리고 후속 절차에 이르는 모든 프로젝트 실행 단계의 기술 적용 기여에 초점을 두고 있습니다.

핵심 인터랙션 프로그래밍

- 게임 오브젝트(Game Object) 동작 및 물리(Physics) 구현 및 설정
- 입력 및 컨트롤 구현 및 설정
- 카메라 뷰(Camara View) 및 동작(Movement) 구현 및 설정

아트 파이프라인(**Art Pipeline**) 작업

- 머티리얼(Material), 텍스처(Texture) 및 셰이더(Shader)에 대해 이해하고, Unity의 렌더링 API와 인터랙션하는 스크립트 작성
- 조명(Lighting)에 대해 이해하고, Unity의 조명 API와 인터랙션하는 스크립트 작성
- 2D 및 3D 애니메이션에 대해 이해하고, Unity의 애니메이션 API와 인터랙션하는 스크립트 작성
- 파티클(Particle) 시스템 및 효과에 대해 이해하고, Unity의 파티클 시스템 API와 인터랙션하는 스크립트 작성

애플리케이션 시스템 개발

- 메뉴 시스템, UI 내비게이션 및 애플리케이션 설정과 같은 애플리케이션의 인터페이스 흐름을 위한 스크립트 해석
- 캐릭터 크리에이터, 인벤토리, 스토어프론트 및 인앱 구매와 같이 사용자가 제어하는 커스터마이징을 위한 스크립트 해석
- Unity 애널리틱스 및 PlayerPrefs와 같은 기술을 활용하여 스코어링, 레벨링 및 게임 내 수익 관리와 같이 사용자가 진행하는 기능을 위한 스크립트 분석
- HUD, 미니 맵 및 광고 등의 2D 오버레이에 대한 스크립트 분석
- 애플리케이션과 사용자 데이터의 저장 및 검색을 위한 스크립트 식별
- 네트워킹 및 멀티플레이어 기능의 영향 인식 및 평가

씬(Scene) 및 환경 디자인을 위한 프로그래밍

- 오디오 에셋 구현을 위한 스크립트 결정
- 게임 오브젝트 인스턴트화, 해체 및 관리를 위한 방법 파악
- Unity 내비게이션 시스템으로 경로를 탐색하기 위한 스크립트 결정

성능 및 플랫폼 최적화

- Unity 프로파일러와 같은 툴을 사용하여 오류 및 성능 문제 검증
- 특정 빌드 플랫폼 및/또는 하드웨어 설정에 대한 요구사항을 해결하기 위한 최적화 방법 파악
- XR 플랫폼에 공통적으로 사용되는 UI 어포던스 및 최적화 결정

전문 소프트웨어 개발 팀 작업

- Unity 콜라보레이트와 같은 기술을 사용하여 버전 관리의 사용 및 영향과 관련된 개념 파악
- Unity 프로파일러 및 기존의 디버깅 및 테스트 기법 등 개발자 테스트와, 개발자 테스트가 소프트웨어 개발 과정에 미치는 영향에 대한 지식 입증
- 모듈성, 가독성 및 재사용성을 위해 스크립트를 구조화하는 기술 파악

인증 시험 주제

핵심 인터랙션 프로그래밍

- 게임 오브젝트와 환경의 동작 및 인터랙션 구현
- 입력 및 컨트롤 구현 방법 파악
- 카메라 뷰 및 동작 구현 방법 파악

아트 파이프라인 작업

- 머티리얼, 텍스처 및 셰이더에 대한 지식—Unity 렌더링 API
- 조명에 대한 지식—Unity 조명 API
- 2D 및 3D 애니메이션에 대한 지식—Unity 애니메이션 API
- 파티클 시스템에 대한 지식—Unity 파티클 API

애플리케이션 시스템 개발

- 메뉴 시스템, UI 내비게이션 및 애플리케이션 설정과 같은 애플리케이션 인터페이스 흐름
- 캐릭터 크리에이터, 인벤토리, 스토어프론트 및 인앱 구매와 같이 사용자가 제어하는 커스터마이징
- Unity 애널리틱스와 같은 툴을 활용하여 스코어링, 레벨링 및 게임 내 수익 관리와 같이 사용자가 진행하는 기능 구현
- HUD, 미니 맵 및 광고 등의 2D 오버레이 구현
- 애플리케이션과 사용자 데이터 저장 및 검색
- 네트워킹 및 멀티플레이어 기능의 가치 및 영향 파악

씬 및 환경 디자인을 위한 프로그래밍

- 오디오 에셋 구현을 위한 스크립트 결정
- 게임 오브젝트 인스턴트화, 해체 및 관리를 위한 방법 파악
- Unity 내비게이션 시스템으로 경로를 탐색하기 위한 스크립트 결정

성능 및 플랫폼 최적화

- Unity 프로파일러와 같은 툴을 사용하여 오류 및 성능 문제 검증
- 특정 빌드 플랫폼 및/또는 하드웨어 설정에 대한 요구사항을 해결하기 위한 최적화 방법 파악
- XR 플랫폼에 공통적으로 사용되는 UI 어포던스 및 최적화 결정

소프트웨어 개발 팀 작업

- 버전 관리: Unity 콜라보레이트와 같은 툴의 사용 및 영향
- 테스트 및 테스트가 소프트웨어 개발 과정에 미치는 영향
- 모듈성, 가독성 및 재사용성을 위해 스크립트를 구조화하는 기술 파악

예제 문항

문항 1

프로그래머가 UI 메뉴 시스템을 구현 중입니다. 각 메뉴는 UI 패널 한 개와 하나 이상의 UI 버튼으로 구성되며, 이 모두가 UI 캔버스(UI Canvas) 오브젝트의 자식 오브젝트가 됩니다. 전체 UI 메뉴 시스템은 추가로 로드되는 별도의 씬에서 생성됩니다.

패널과 버튼의 아트 스타일은 색(Color), 텍스처(Texture), 버튼 전환 방식(Button Transition Type) 등에 있어 일관성이 필요한데, 아트 디렉터가 이 스타일을 아직 결정하지 못했습니다. 아트 디렉터는 이러한 설정을 프로그래머의 UI 작업과 동시에 진행하려고 합니다. 이럴 경우 아트 디렉터가 스타일을 변경한다면 씬의 기존 오브젝트와 신규 오브젝트 전체에 영향을 미치게 됩니다.

프로그래머가 Unity의 기능을 사용하여 기능적인 메뉴 시스템을 간편하게 만들면서, 아트 디렉터가 디자인(look and feel) 작업을 동시에 독립적으로 진행할 수 있도록 요건을 충족할 수 있는 최선의 방법은 무엇입니까?

A `UI.Button` 및 `UI.Panel`에 대한 서브 클래스를 만들고 디자인 값을 프로그래밍하여 설정한다.

B 새 버튼과 패널 머티리얼(Material)을 만들어 이를 씬(Scene)의 모든 버튼과 패널에 할당한다.

C 버튼 및 패널에 프리팹(Prefab)을 사용하고, 아트 디렉터가 프리팹을 수정하게 한다.

D 아트 디렉터의 입력에 따라 씬 파일에서 값을 검색하여 바꿀 수 있도록 스크립트를 작성한다.

문항 2

3D 무한 러닝(endless runner) 게임이 철도 차량 기지에서 병렬로 배치된 여러 선로를 배경으로 설정되어 있습니다. 플레이어는 항상 선로에서 앞을 향해 달리며, 마주 오는 열차를 뛰어넘거나 인접한 선로 위로 점프하여 건너가는 방식으로 열차를 피해야 합니다.
선로에 새로 추가되는 각 열차는 그 선로에 있는 다른 모든 열차 뒤에 추가됩니다. 그러나 플레이어를 향해 이동하는 열차는 속도가 일정하지 않거나, 전혀 움직이지 않는 경우도 있으므로 열차끼리 겹치는 상황이 종종 발생하여 수정이 필요합니다.

새 열차가 이미 같은 선로에 있는 열차와 겹치지 않도록 하기 위해 성능 면에서 가장 적절한 방법은 무엇입니까?

A 선로상에 열차를 생성할 때 선로의 맨 뒤에 있는 열차와 새 열차의 속도, 그리고 선로의 맨 뒤에 있는 열차가 플레이어를 지나갈 때 사라지는(de-spawn) 지점을 이용하여 문제를 방지할 수 있는 생성 포지션을 결정한다.

B 열차가 움직이고 있을 때 열차의 앞부분에서 정방으로 레이를 캐스트하고, 레이와 충돌하는 열차는 더 빠른 열차의 속도로 전방으로 밀어낸다.

C 선로상에 열차를 생성할 때 리지드바디를 추가한 후 물리적인 힘으로 열차를 움직이게 한다.

D 선로상에 열차를 생성할 때 열차의 속도와 비례하는 길이를 가지는 박스캐스트(BoxCast)를 이용하여 그것이 카메라 뒤에 위치할 때까지 다른 열차와 충돌하지 않게 한다.

문항 3

프로그래머가 어둡고 음산한 방을 배경으로, 벽, 바닥 및 천장에 으스스하게 일렁이는 그림자를 드리우며 깜박거리는 횃불을 만들고 있습니다. 프로그래머는 횃불에 연결된 MonoBehaviour에 다음 함수를 작성합니다.

```
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);
}
```

런타임 시 횃불에서 광원이나 그림자를 드리우지 않습니다. Unity 에디터에는 광원이 기본값으로 설정되어 있습니다.

이 코드가 요구하는 대로 작동하게 하려면 프로그래머는 무엇을 변경해야 합니까?

A light.lightBakeType을 LightmapBakeType.Realtime으로 설정

B light.range를 10으로 설정

C light.shadows를 LightShadows.Hard로 설정

D light.type을 LightType.Point로 설정

문항 4

플레이어가 광물을 찾기 위해 땅을 팔 수 있는 채굴 시뮬레이션(mining simulation) 게임을 프로그래머가 개발 중입니다. 한 현장에서 플레이어는 기존의 동굴 시스템을 가로지르는 터널을 만들 수 있습니다. 설계 문서에 따르면 기존의 동굴과 새 터널에서 발생하는 오디오에 약간의 리버브가 들어가야 합니다. 프로그래머는 사용자가 가장 가까운 동굴의 리버브 구역(ReverbZone)에 지속적으로 있도록 해야 합니다.

이 요구사항을 충족하기 위해 프로그래머는 AudioReverbZone 프로퍼티를 어떻게 조정해야 합니까?

A 새 영역에 맞게 반향을 높인다.

B 두 리버브 존의 maxDistance를 높여 새 연결 영역 안에서 만나도록 한다.

C 새 영역을 수용하도록 리버브를 높인다.

D 새 영역의 decayTime을 높인다.

문항 5

프로그래머가 로딩 함수를 작성하는 동안 컴파일 오류가 발생했습니다.

```
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;
    }
}
```

이 오류를 해결하려면 프로그래머는 어떻게 해야 합니까?

A `yield return null`을 `yield return WaitForSeconds(0)`로 변경

B `void LevelLoading()`을 `IEnumerator LevelLoading`으로 변경

C `SceneManager.LoadSceneAsync("Level _ 01")`를 `Application.LoadLevelAdditiveAsync("Level _ 01")`로 변경

D `while (!async.isDone)`을 `while (!async.allowSceneActivation)`으로 변경

문항 6

주행 게임에서 수평 입력 축이 스티어링을 제어하도록 입력 시스템이 매핑되었습니다. 테스트 중에, 일부 조이스틱 장치에서 스틱이 중앙에 위치해 있는데도 스티어링 입력이 등록되는 문제가 발견되었습니다.

이 문제를 해결하려면 입력 시스템의 축에 어떤 변경을 해야 합니까?

A 중력값(Gravity)을 높인다.

B 스냅(Snap)을 true로 설정한다.

C 데드존(Deadzone)을 늘린다.

D 감도(Sensitivity)를 낮춘다.

문항 7

외계 행성을 배경으로 하는 어드벤처 게임에서 플레이어는 다양한 생명체를 퇴치해야 합니다. 하나씩 없앨 때마다 플레이어의 점수가 올라갑니다. 설계 문서에는 이 점수를 플레이어의 계정에 연결하여 나중에 플레이어가 다른 플레이 세션에 있거나 다른 장치를 사용하는 경우에도 해당 점수를 가져올 수 있게 해야 한다고 명시되어 있습니다.

점수 데이터를 저장하기 위해 프로그래머가 사용할 수 있는 가장 좋은 방법은 무엇입니까?

A 점수 데이터를 보유한 게임 오브젝트에 `DontDestroyOnLoad()`를 사용하고 애플리케이션 종료 직전에 데이터를 서버에 업로드한다.

B 점수가 업데이트될 때마다 `PlayerPrefs`에 저장하고 애플리케이션 종료 직전에 서버에 업로드한다.

C 정적 값(static value)을 사용해 점수 데이터를 저장하여 다음 플레이 세션에서 사용할 수 있게 한다.

D 데이터 직렬화(data serialization)를 사용하여 점수 데이터를 지속적으로 저장하고 서버에 업로드한다.

정답: C, A, D, B, B, C, D