unity

unity

Certified

Expert

Technical Artist: Rigging & Animation

시험 목표

Unity 인증 전문 테크니컬 아티스트: 리깅 및 애니메이션

© 2017 Unity Technologies

역할

리깅 및 애니메이션 테크니컬 아티스트는 게임 아티스트와 프로그래머 사이의 중요한 다리가 되어 예술적 비전과 플랫폼 제한을 존중하면서도 게임에 체계적으로 통합될 수 있는 에셋을 제공합니다. 테크니컬 아티스트 에셋은 변화하는 앱 요구 사항과 플레이 환경에 적응할 수 있을 만큼 유연하며, 발전하는 사용자 요구에도 대응할 수 있습니다. 아티스트의 깊이 있는 리깅 및 애니메이션 기술은 애니메이션, 게임 개발 및 캐릭터 제작을 지원하며 게임 제작 파이프라인을 향상시킵니다.

또한 게임 개발팀은 리깅 및 애니메이션 테크니컬 아티스트의 핵심 기술을 사용해서 오디오 및 애니메이션을 포함한 복합 애니메이션과 게임 오브젝트(GameObjects) 스크립트를 작성합니다. 아티스트는 다양한 플랫폼에서 이러한 에셋을 최적화하는 파이프라인 도구를 만들고 지원하며 이 과정이 게임의 기술 설계 문서(TDD)를 준수하는지 확인합니다.

이 업무의 직함

- 테크니컬 아티스트
- 리거
- 테크니컬 애니메이터
- 테크니컬 캐릭터 디자이너

© 2017 Unity Technologies

필수 조건

이 전문가 인증 시험은 이 분야에서 수년 동안 일했으며, 다음과 같이 다양하고 실용적인 고급 응용 프로그램 경험을 쌓은 사람들에게 권장됩니다.

- 비디오 게임 개발 스튜디오 경험(출시된 타이틀 두 개 이상 포함)
- C++, C# 또는 Unityscript 언어로 스크립팅/코딩에 대한 해박한 지식
- 초기 게임 컨셉부터 출시까지 게임의 전체 주기 경험
- 리깅/캐릭터 설정 및 애니메이션 경험
- Python, MEL, MaxScript와 같은 디지털 콘텐츠 제작(DCC) 스크립팅 언어에 익숙
- 캐릭터 및 환경 설정을 포함한 게임 애니메이션 파이프라인에 대한 이해
- 파일 구조, 작명 규칙 및 확립된 프로토콜에 대한 믿음직한 체계화 기술
- Adobe Creative Suite, Substance Designer, Substance Painter, Quixel Suite, Autodesk Maya 및 3ds Max, Pixologic ZBrush, Motion Builder 등 에셋 제작 도구의 능숙한 사용

© 2017 Unity Technologies

핵심 기술

전문 테크니컬 아티스트: 리깅 및 애니메이션 인증 시험은 지원자가 리깅 및 애니메이션 에셋을 게임에 효과적으로 통합하는 데 필요한 기술을 갖추고 있는지 확인합니다. 합격한 지원자는 다음 분야에서 고급 능력을 갖추게 될 것입니다.

프로토타입 제작

- 게임 설계 문서(GDD)를 평가하여 나머지 설계 팀이 게임을 만들고 '설계대로 진행'할 수 있는 애니메이션 중심의 Editor 도구 결정
- 프로토타입을 제작 및 평가하여 모범 사례를 개발하고 플랫폼 성능 사양별 기술 설계 문서(TDD) 개선
- 애니메이션 및 리깅 문제에 대한 기술 솔루션 평가 및 추천

파이프라인 제작

- 에셋 가져오기를 사용자 지정 및 자동화
- 게임 오브젝트(GameObjects)의 속성을 순서대로 수정 및 변경
- 복수의 게임 오브젝트(GameObjects) 파라미터 제어
- 행동 및 애니메이션을 순서대로 구현

게임 오브젝트(**GameObjects**) 준비

- 게임 구현을 위해 디테일 레벨(LOD)을 통한 에셋 프리팹(Prefab) 준비
- 휴머노이드 및 일반 리그 애니메이션 유형 구현 및 매핑
- 프리팹(Prefab)으로 배치하기 위해 조인트(Joint), 클로스(Cloth), 리지드바디(RigidBody), 물리(physics) 컴포넌트를 사용해서 복합 어셈블리를 리그 및 스크립트
- 원활한 게임플레이를 위한 커스텀 물리 머티리얼 제작 및 시험
- 복합 메시 콜라이더(Mesh Collider) 및 물리 머티리얼 평가 및 최적화

© 2017 Unity Technologies

애니메이션 준비

- 스테이트 머신(State Machine)에서 블렌드 트리(Blend Tree) 제작
- 스테이트 머신(State Machine)에서 복합(활성 상태의 다중 레이어) 및 고성능 행동 스크립트
- 스테이트 머신(State Machine) 레이어로 레이어 애니메이션 제작
- 스테이트(State) 및 행동 제작으로 블렌드 모양(Blend Shape)에 작용 및 트랜지션(Transition)
- 애니메이션 클립에서 프레임 기반의 오디오/FX 설정

성능 및 최적화

- 대상 플랫폼의 사양 및 제약 이해
- FK 및 IK 리그의 차이점과 상대적인 성능 영향 이해
- 각 플랫폼 요구 사항별 복합 어셈블리 테스트 및 최적화
- 프로파일러를 사용하여 씬(Scene) 성능을 평가하고 병목현상 확인
- 리그 복잡도, 일괄 처리, 버텍스 쉐이더(Vertex Shader) 메소드에 대한 CPU 및 GPU 최적화 평가

© 2017 Unity Technologies

인증 시험 주제

도구 및 파이프라인

- Editor 사용자 지정
- 에셋 사용자 지정
- 커스텀 도구를 사용한 프로세스 자동화

프로토타입 제작

- 리깅 및 애니메이션 프로토타입 제작

애니메이션 및 리깅

- 애니메이션 시스템 스테이트 머신(State Machine) 및 애니메이션 이벤트 구성
- 리그 설정 및 애니메이션
- 동적 애니메이션을 위한 물리 컴포넌트

성능

- 씬(Scene) 최적화

© 2017 Unity Technologies

예제 문항

문항 1

전투 게임에 대한 게임 설계 문서(GDD)에서 명시한 바에 따르면 논플레이어 캐릭터(NPC)는 대기 상태(idle), 달리기, 공격, 방어 등 다양한 애니메이션을 포함합니다. NPC들은 전투에서 무장하고 있으며 양손 검을 사용합니다.

게임 설계 문서(GDD)는 NPC가 게임에서 다양한 모습을 구현하기 위해 포즈 레이어(Layer)에 여러 가지 포즈를 보유하도록 명시하고 있습니다. 야수(Brute) 포즈 애니메이션 클립에서 캐릭터는 어깨를 앞으로 내밀고 등을 구부립니다. 영웅(Hero) 포즈 애니메이션 클립에서는 캐릭터가 위풍당당하게 어깨와 가슴을 활짝 폅니다. 게임 테스트 시 포즈 레이어가 적용되면 NPC가 양손으로 검의 손잡이를 일직선으로 잡지 **않**습니다.

이 문제의 해결 방법은 무엇입니까?

A OnAnimatorIK()를 사용해서 양손을 무기로 옮길 때의 위치, 회전 및 관련 가중치를 설정한다.

B 각 아바타마다 가슴에 근육별 설정(Per-Muscle Settings)을 사용해 양손이 적절하게 움직이도록 만든다.

C OnStateMove()를 오버라이드하는 StateMachineBehaviour를 사용해서 animator.MatchTarget()을 호출하여 손의 위치를 조정한다.

D 각 아바타마다 양손으로 무기를 가까이 또는 멀리 잡을 수 있도록 추가 뼈대(Optional Bones)를 설정한다.

© 2017 Unity Technologies

문항 2

게임 설계 문서(GDD)에서 다음 요구사항을 명시하고 있습니다.

- 무기는 양손을 바꿔서 잡을 수 있어야 합니다.
- 무기는 PropWeapon이라는 조인트(Joint)의 부모가 되어야 합니다.
- PropWeapon은 캐릭터 루트의 자식(Child)이 됩니다.

디지털 콘텐츠 제작(DCC) 패키지에서 애니메이션 팀은 PropWeapon 조인트(Joint)를 제한하여 무기를 잡는 손을 바꿀 수 있도록 만들었습니다. 애니메이션의 모든 프레임을 베이크하고 FBX를 사용하여 익스포트하였습니다. 테크니컬 아티스트는 Editor에서 게임을 플레이할 때, 무기가 흔들리지만 디지털 콘텐츠 제작(DCC) 패키지에서는 표시되지 않는다는 것을 발견합니다.

이 흔들림 현상의 원인은 다음 중 무엇입니까?

A 반대쪽 손에서 재생되는 애니메이션이 PropWeapon 조인트(Joint)에 영향을 미친다.

B 압축 설정의 위치 오류가 너무 낮다.

C 무기에 부착된 조인트(Joint)가 손의 직접적인 부모가 되지 않았다.

D 애니메이터의 루트 모션이 PropWeapon 조인트(Joint)의 위치에 영향을 미친다.

© 2017 Unity Technologies

문항 3

서바이벌 게임에 대한 게임 설계 문서(GDD)에는 보안 로봇에게 쫓기는 인간 플레이어 캐릭터가 등장합니다. 로봇의 외골격은 견고하며, 주 관절에 유압 피스톤이 드러나 있습니다. 애니메이션 디렉터는 로봇이 더 뻣뻣하고 기계적으로 동작하게 해 달라고 요청했습니다. 인간 및 로봇 캐릭터에는 모든 이동(locomotion) 애니메이션을 재사용하기 위한 휴머노이드 리그 애니메이션 타입(Humanoid Rig Animation Type)이 있습니다.

모션 스타일의 차이를 강조하려면 테크니컬 아티스트는 캐릭터의 아바타를 어떻게 설정해야 합니까?

A 근육별 설정(Per-Muscle Settings)을 통해 로봇의 모션 범위를 제한하고, 인간 설정은 기본값으로 둔다.

B 로봇 아바타를 A-포즈, 인간 아바타를 T-포즈로 설정한다.

C 로봇의 위쪽 팔과 위쪽 다리의 뼈대를 길게 늘이고, 인간의 위쪽 팔과 위쪽 다리의 뼈대는 제작한 그대로 둔다.

D 로봇의 추가 뼈대(Optional Bones)를 사용해서 늘어난 모션 범위를 표시하고, 인간의 추가 뼈대(Optional Bones)는 비활성화한다.

© 2017 Unity Technologies

문항 4

테크니컬 아티스트는 휴머노이드 캐릭터가 움직일 때 환경에 정확하게 발을 디디는 시스템을 개발하고 있습니다. 이 솔루션은 다양한 크기의 캐릭터에 적용됩니다. 이 솔루션에는 두 개의 충돌 메시(Collision Mesh)가 있습니다. 휴머노이드 캐릭터 캡슐의 메시는 움직임을 주고, 더 정확한 다른 하나의 메시는 발을 땅에 디디게 합니다.

시스템 중 일부에서는 지면에 발의 위치를 투사해야 하며, 다음과 같습니다.

```
  Vector3 ProjectPositionOnGround(Vector3 position)
   {
        Vector3 ret = position;

        RaycastHit hitInfo = new RaycastHit();
        if (Physics.Raycast(position + new Vector3(0, 0.5f, 0), new
Vector3(0, -1, 0), out hitInfo, 1.0f, m_LayerMask))
        {
            ret = hitInfo.point;
        }

        return ret;
  }
```

테크니컬 아티스트는 일부 캐릭터에서 IK 플랜팅이 예상대로 작동하지 않는다는 것을 발견했습니다. 이 문제를 해결하려면 코드를 어떻게 수정해야 합니까?

A 모든 캐릭터가 적절한 충돌 메시(Collision Mesh)에 대해 레이캐스팅(Raycasting)되도록 레이어를 동적으로 설정한다.

B 캐릭터 크기에 따른 오프셋을 사용하여 레이캐스트 오리진(Raycast Origin)에 변화를 준다.

C 레이캐스트(Raycast) 방향 벡터의 크기를 조정하여 항상 콜라이더(Collider)에 닿는지 확인한다.

D 레이캐스트 오리진(Raycast Origin) 및 maxDistance 값을 위해 캐릭터 크기에 따른 오프셋을 사용한다.

© 2017 Unity Technologies

문항 5

높고 뻣뻣한 깃이 달린 외투를 입은 캐릭터 디지털 콘텐츠 제작(DCC) 패키지에 리그를 설정하여 애니메이션 도중 깃이 머리와 턱을 관통하지 못하도록 합니다. 원하는 표현을 구현하기 위해 물리에 영향을 받지 않는 12개의 추가 뼈대를 깃 설정에 사용합니다. 게임 내 모든 캐릭터는 휴머노이드 리그(Humanoid Rig)로 임포트했으며 동일한 애니메이션 세트를 재사용합니다.

애니메이션 데이터가 타겟 빌드 플랫폼의 공간을 너무 많이 차지합니다.

기존 애니메이션 행동과 일치하면서도 애니메이션을 최적화하기에 가장 효율적인 방법은 무엇입니까?

A 디지털 콘텐츠 제작(DCC) 리그를 편집하고 외투 깃의 조인트(Joint) 개수를 줄인다.

B 디지털 콘텐츠 제작(DCC) 리그를 편집하고 외투 깃의 조인트(Joint) 설정을 BlendShapes로 대체한다.

C 애니메이션을 임포트할 때 12개의 외투 깃 뼈대를 마스크 처리한 후, Unity Physics 컴포넌트로 깃 충돌(Collision) 행동을 다시 생성한다.

D 애니메이션을 임포트할 때 12개의 외투 깃 뼈대를 마스크 처리한 후, 디지털 콘텐츠 제작(DCC) 리그 설정과 일치하는 행동을 구현하는 스크립트를 생성한다.

정답: A, B, A, D, D

© 2017 Unity Technologies