**개발 결과 보고서**

2021182043 홍륜기

2021180042 한진우

|  |  |  |
| --- | --- | --- |
| **제목** | **3D 환경으로 재해석한 종 스크롤 슈팅** | |
| **소개** | **캐릭터를 조종하여 다가오는 적들을 피하고, 무찔러라**   * 3D 환경으로 재해석한 고전적 종 스크롤 슈팅 게임 * 좌/우 방향으로 플레이어 캐릭터를 움직이며, 전방으로 총알을 발사 * 주기적으로 생성되어 날아오는 적들 * 캐릭터가 적에게 닿지 않도록 능동적인 컨트롤 필요 | |
| **조작** | **키보드 사용** |  |
| 방향키(←/→) | : 플레이어 캐릭터 좌/우 이동 |
| 스페이스 | : 총알 발사 |
| Q | : 게임 종료 |

|  |  |  |  |
| --- | --- | --- | --- |
| **구조** | **협업 관련** | **GitHub을 이용한 협업**  작업간 파일 충돌을 최소화하기 위해  커밋 규칙 관련 사전 합의 진행 (커밋 주기, 주의사항 등…) | |
| **시스템 구조** | **• 로직 진행**  **텍스트, 원, 그래픽, 폰트이(가) 표시된 사진  자동 생성된 설명** | |
| **• Object 클래스**  게임 내 사용되는 오브젝트들의 형태.  각 오브젝트(Player, Enemy, Bullet 등)들은 이 형태를 따름. | |
| init() | : 초기화 함수. 오브젝트 생성 시 실행함. |
| update() | : 매 업데이트(프레임) 마다 수행할 작업. |
| handle\_events() | : 각종 유저 조작을 처리(glutKeyboardFunc 등) |
| get\_bb() | : 충돌 범위를 구하는 함수. |
| handle\_collision() | : 충돌 시 처리를 담당하는 함수. |
| take\_damage() | : 피해 처리 함수 |
| add\_effect() | : 이펙트 추가 함수 |
| remove\_effect() | : 이펙트 제거 함수 |
| **• 충돌 처리**  **1. 오브젝트 생성 시 충돌 그룹 지정**  - 맵 자료구조 형태로 collision\_pair를 구성함.  - (“group”, object\_1, object\_2) 형태.  - 오브젝트 생성 시 오브젝트를 해당하는 collision\_pair에 추가  **ex) Enemy 클래스 오브젝트, new\_enemy 생성 시:** | |

|  |  |  |
| --- | --- | --- |
| **구조** | **시스템 구조** | **2. 같은 그룹 오브젝트 충돌 검사**  - ex) “ally\_bullet:enemy” 그룹 내 ally\_bullet들과 enemy간 검사.  - AABB 방식을 3차원 공간으로 확장시킨 방식을 사용함. |
| **3. 각 오브젝트의 handle\_collision() 함수 호출**  - 각 오브젝트가 각자의 handle\_collision()을 처리하는 게 원칙.  - ex) Enemy.handle\_collision() |

|  |  |  |
| --- | --- | --- |
| **구조** | **시스템**  **구조** | **• 배경 스크롤링**  - 커다란 원통을 회전시킴.  - 원통 주위 일정 범위에 직육면체를 배치하여 건물 형태 구현    좀 더 멀리서 바라본 게임 화면. |
| **• 오브젝트 생성 / 삭제에 오브젝트 풀링 구현**  - 총알 등 자주 사용되는 오브젝트 생성/삭제 메커니즘 개선  - 오브젝트 풀 시스템을 이용해 생성/삭제 비용 최적화  - 1. 미리 오브젝트를 일정량 생성하여 풀(pool)에 저장  - 2. 생성이 필요할 때 풀에서 오브젝트를 가져오고  - 3. 사용을 마친 오브젝트는 풀에 반환 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **구조** | **시스템 구조** | **• Effect 시스템 - 1**  - 오브젝트에 적용되는 효과들을 Effect 클래스 형태로 구현.  - apply(): 효과가 적용될 때 수행  - remove(): 효과가 끝날 때 수행  - update(): 효과 지속시간 도중에 수행 | | | |
| **• Effect 시스템 - 2**  - 각 오브젝트는 자신에게 적용된 Effect들을 관리함.  - 오브젝트에 적용되는 각종 효과들을 효율적으로 관리 가능  ex) 이동속도 감소, 지속 피해, 공격력 증가 등 | | | |
| **구조** | **렌더링 구조** | **•** 뒷 배경을 만들 배경 큐브 구현  - 회전하는 큐브 안에 들어감  - 큐브 내부에 텍스처 매핑해 뒤 배경 구현    큐브 밖에서 바라본 화면 | | | |
| **•** 텍스처 파일 로딩  - 각 배경 모델은 자신이 사용할 텍스처를 객체 생성시 LoadTexture()로 읽어 보관함 | | | |
| **진행상황** | **시스템 구현** | 플레이어 조작 | | : | 이동, 총알 발사 등 플레이어 조작 구현 완료 |
| 전투 | | : | 충돌 처리, HP 증감 및 사망 시스템은 구현  완료하였으나, 적 행동을 제대로 구현하지 못함. |
| 스테이지 | | : | 기승전결이 갖춰진 하나의 게임 플레이 루프 형태를 갖추지 못함. |
| **렌더링 구현** | 조명 | | : | 노멀 벡터의 적용 오류로 제대로 구현하지 못함 |
| 텍스처 | | : | 하늘, 도시 정경 등 배경 텍스처 구현 |
| 스크롤링 | | : | 배경 회전을 통한 밤낮 전환 시스템 구현 |
| **소감 및**  **후기** | **홍륜기**  **(시스템 구현)** | | 재미있었다. 개발 과정에서 생각지 못한 문제가 생길 때도 있었지만, 하나하나 해결해 가며 결국 좋은 경험이 되었던 것 같다. 이번 경험을 발판 삼아 다음에는 더 좋은 결과물을 얻을 수 있으리라 생각한다. | | |
| **한진우**  **(렌더링 구현)** | | 최근에 배운 익숙하지 않은 조명과 텍스쳐링을 중점으로 맡았다. 노말 좌표 설정에 오류가 있어 조명 효과를 잘 처리하지 못한 건 아쉬운 일이지만 다음에는 더 잘할 수 있으리라 생각한다. | | |