**프로젝트 기술문서**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **프로젝트 명** | Evilness | | | |
| **프로젝트 기간** | **프로젝트 인원** | | | |
| 2017-03월 ~ 11월  (약 8개월간) | 프로그래머 | 기획 | 그래픽 | 합 계 |
| 2명 | 2명 | 11명 | 15명 |
| **개발 도구** | Unity 3D, C# | | | |
| **상세 내용** | | | | |
| **\* 프로젝트 소개**  - Evilness는 Unity3D 엔진으로 제작 되었고, 졸업작품 게임으로 시원한 전투와 화려한 이펙트가 있는 3D 액션게임으로, 적들을 무찌르며, 최종 보스를 쓰러트리고 게임을 클리어하는 게임입니다. 2017년 졸업 작품으로 진행된 프로젝트 입니다. 3월 초 학기가 시작되는 기점으로 총 15명의 개발자가 프로젝트에 참여를 하게 되었습니다. 다른 팀과 비교하였을 때 그래픽과 프로그래머가 가장 많은 팀이고, 최고의 퀄리티를 보여주었다고 생각합니다. | | | | |
| **스크린샷** | | | | |
|  | | | | |
| **\* 프로젝트에서 맡은 역할**  - Evilness 프로젝트에서 메인 프로그래머를 맡고 있으며, 시스템 기획자와 상의하여 게임 로직을 설계하였습니다. AI를 주로 하고 있으며, AI의 FSM을 제작하고, 보스몬스터 패턴을 제작하였고, Unity3D의 Nav Mesh Agent를 사용하지 않고 NavMesh의 Bake된 Map정보를 가지고 새로운 AI Nav Mesh Agent를 만들어 사용하였습니다. 또한, Game Manager를 통해 게임에 룰을 제작하고 있고 캐릭터 스크립트의 기반과 캐릭터의 스킬를 만들었습니다.  **\* 프로젝트 진행 시 있었던 문제 해결 및 진행 과정**  - AI를 어떻게 하면 서로가 뭉치지 않고 서로가 피해가면서 플레이어를 공격하도록 구현을 하고 싶었습니다. 처음에는 Unity에 Nav Mesh Agent를 사용했지만 목적지에 도달할 경우 서로가 밀어내는 현상이 있어 Obstacle 스크립트를 만들어서 이 스크립트를 가지고 있는 객체랑 일정 거리가 되면 피해가도록 제작을 하였습니다. 캐릭터가 보도블럭을 올라가는 것에도 문제가 있었는데, 레이캐스트를 써서 보도블럭이 탐지가 되면 살짝 캐릭터를 띄워서 올라가도록 만들었습니다. 또, 액션게임이기 때문에 무엇보다 자연스러운 캐릭터 애니메이션이 중요했습니다. 더욱 실감나게 제작하려고 노력했습니다.  **\* 유튜브 링크 :** [**https://youtu.be/ITK0-riGupo**](https://youtu.be/ITK0-riGupo) | | | | |
| **<클래스 다이어그램>**  매니저를 자주 사용 하기 때문에 싱글톤 클래스를 만들어 매니저에 상속을 시켜서 사용하고 있습니다. 전략디자인 패턴으로 Unit의 종류를 관리를 하고 무기의 종류도 관리하고 있습니다. AIControll에서는 몬스터의 종류에 따라 행동패턴만 다르게 구현할 수 있게 상속받아서 제작하였으며 모든 움직임은 PlayerManager, AIManager에서 관리를 하고 있습니다.    **<Obstacle 장애물 체크>**  장애물 클래스를 만들어서 타입을 나누고 그 타입에 맞춰 얼마만큼 범위를 피해나갈 것인가를 계산했습니다. ObstacleManager에서는 모든 장애물 스크립트를 가지고 있는 객체를 저장해두고 항시 비교를 해주며 AI가 장애물 스크립트를 가지고 있는 모든 객체들을 스스로 피해갈수 있도록 만들어주었습니다. | | | | |

|  |
| --- |
| **<카메라 뒤 장애물 있을 시 확대 축소>**  플레이어를 따라 카메라가 움직이게 될 때 플레이어가 장애물 근처에 갔을 경우 장애물을 뚫고 들어가 캐릭터를 가리는 현상이 있었습니다. 다른 게임들을 해보면서 그 게임들은 어떤 형식으로 카메라를 처리를 했는가를 많이 살펴봤습니다. 플레이어에서 Ray Cast를 사용해서 카메라까지 Ray를 사용해 체크를 하고, Ray에 Hit가 된 정보에서 Tag에 Maps(모든 장애물) 또는 Ground(바닥)일 경우에 카메라의 포지션을 Ray가 닿은 Hit Point까지 Lerp로 서서히 움직이게 만들었습니다. 또, Ray가 태그 Maps 또는 Ground에 닿지 않을 경우에는 다시 원래 있던 포지션으로 돌아오게 제작하였습니다.  C:\Users\user\AppData\Local\Microsoft\Windows\INetCache\Content.Word\21.png  **<CUnit으로 공통스텟을 생성>**  플레이어와 몬스터는 공통으로 갖는 스텟을 상속받아서 선언을 할 수 있게 설계하였고, 플레이어 및 몬스터만에 가지고 있는 속성을 따로 만들어주고 코드를 간편하게 정리가 되었습니다. 여러 몬스터가 나올 수도 있고, 플레이어에게 여러가지의 검이 나올 수 있을 가능성을 가지고 전략디자인패턴을 사용하여 제작하였습니다. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **프로젝트 명** | Count Slime | | | | |
| **프로젝트 기간** | **프로젝트 인원** | | | | |
| 2016-08월 ~ 09월  약 1개월 | 프로그래밍 | 기획 | 그래픽 | 합 계 | |
| 1 | 2 | 4 | 7 | |
| **개발 도구** | Unity 3D, C# | | | | |
| **상세 내용** | | | | | | Unity3D C# |
| **\* 프로젝트 소개**  - Count Slime 프로젝트는 아기자기 하고 귀여운 슬라임을 오크들로부터 지켜나가면서 스테이지를 클리어하는 디펜스 게임입니다. 웨이브 마다 다양한 몬스터들이 나오며, 슬라임들은 농장안에서 활발하게 놀다가 체력이 달면 배고픔을 느껴 주변 나무수액을 마시면서 체력을 회복하고 다시 농장으로 돌아와 활발하게 놀수 있는 귀여운 게임입니다. | | | | | |
| **스크린샷** | | | | | |
| C:\Users\Dawony\Desktop\noname01.jpgC:\Users\Dawony\Desktop\noname02.jpg  C:\Users\Dawony\Desktop\noname04.jpgC:\Users\Dawony\Desktop\noname03.jpg | | | | | |
| **\* 프로젝트에서 맡은 역할**  - Count Slime 프로젝트에서 제가 맡은 분야는 프로그래밍 분야를 맡아 진행하였습니다. 전체적인 구조를 만드는 것부터 시작 하였고 전략 디자인 패턴을 이용하여 캐릭터, 몬스터, 슬라임을 관리 하였습니다. 캐릭터에서는 무기 교체가 있기에 쉽게 바꾸기 위하여 Weapon이라는 클래스를 만들어 관리 하였습니다. 몬스터는 Unity3D Nav Mesh Agent를 사용하였습니다.  **\* 프로젝트 진행 시 있었던 문제 해결 및 진행 과정**  - 프로젝트를 진행 할 때 클래스설계를 어떻게 할지 고민을 하다 수업시간에 배운 전략디자인패턴을 사용해보자 라는 생각이 들어 전체적인 기본 틀은 전략디자인패턴을 이용했고, 이 프로젝트를 진행하면 가장 힘들었던 부분은 슬라임이 농장 내에서 자유롭게 움직이다가 HP가 50이하로 떨어졌을 때 나무수액을 먹어서 회복하는 부분에서 많은 오류가 있었지만 코루틴을 잘못 사용 했던 부분을 고쳐서 해결했습니다. 그 이후 코루틴은 어떤 기능을 하는지 조금 살펴보는 기회가 되었습니다.  **\* 프로젝트를 마무리 하면서 느낀 점**  - 짧은 개발 기간에 맞춰서 남들보다 창의성이 있는 것을 찾는 것이 생각보다 쉽지가 않다는 것을 느끼게 되었습니다. 팀원이 일정에 늦어진다고 무작정 재촉하는 것보다, 격려하며 팀원을 믿고 기다리는 것이 가장 중요한 것이라고 다시 한번 생각하게 되었습니다.  **\* 유튜브 링크 :** [**https://youtu.be/j7gS0oNLZec**](https://youtu.be/j7gS0oNLZec) | | | | | |
| **<클래스 다이어그램>**  저는 공통으로 들어갈 것을 생각하고 상속을 사용하며 가상클래스로 만들어서 재정의를 하여서 사용을 했습니다. 그리고 그 안에서 고유의 능력들은 상속받은 클래스 내에서 기능을 구현하였고 이렇게 구현을 하면 관리하기도 쉽다고 판단하였습니다.  **<전략디자인패턴을 이용한 모든 객체 기본 베이스 관리>**  각 기본적으로 공통으로 들어가는 HP, Damage같은 경우에선 CPlayer라는 공통클래스를 생성해주었습니다.  HP가 변화하는 것과 무기 마다 공격 사운드가 다르게 나와야 해서 CWeapon이라는 클래스를 만들어준 후 무기 클래스를 여러 개를 만들어서 관리 했습니다. | | | | | |
| **<공통 Trigger처리>**  클래스 구조가 모두 상속으로 처리를 했기에 충돌처리도 공통으로 만들었습니다. 템플릿을 사용하여 어떤 클래스가 들어오지는 파악하게 만들어서 각자의 상황에 맞게 데미지를 주었습니다.    **<무기 변경 스크립트>**  플레이어가 무기를 습득할 경우 ItemSet 클래스에서는 떨어진 무기랑 충돌이 되었을 경우 그 무기의 정보로 변경을 시켜주는 클래스입니다. WeaponChange 클래스는 떨어진 무기로 변경 시켜주는 데 실제로 이미지가 변경 되는 부분은 CopyComponent에서 플레이어가 가지고 있는 무기랑 떨어진 무기에 있는 Mesh랑 Material를 변경 해주어서 이미지 적으로 변경을 시켜 주었습니다. | | | | |

|  |  |  |
| --- | --- | --- |
| **프로젝트 명** | Dragon Flights | |
| **프로젝트 기간** | **프로젝트 인원** | |
| 2015-11월 ~ 2015-12월  약 1개월 | 프로그래밍 | 합 계 |
| 1 | 1 |
| **개발 도구** | Visual Studio 2015, Win API, C++ | |
| **상세 내용** | | |
| **\* 프로젝트 소개**  - Dragon Flights는 Win API로 제작되었고 카카오게임에 있는 드래곤플라이트를 따라 제작했습니다. 1학년 2학기 윈도우 프로그래밍 수업 시간에 과제물로 제작한 프로젝트입니다. 드래곤플라이트를 비슷하게 제작하려고 노력했고, 각종 아이템과 골드를 먹어서 더욱 강해져서 앞으로 나아가는 종스크롤 액션 게임입니다. | | |
| **스크린샷** | | |
| C:\Users\Dawony\Desktop\사본 -1.jpgC:\Users\Dawony\Desktop\사본 -2.jpgC:\Users\Dawony\Desktop\사본 -3.jpgC:\Users\Dawony\Desktop\사본 -4.jpg | | |
| **\* 프로젝트에서 맡은 역할**  - Dragon Flights는 1인 개발로서 모든 시스템을 제작하였습니다. 드래곤플라이트를 수도 없이 게임을 해보면서 기획을 하였고, 싱글톤 게임매니저를 만들어 상태에 따라 관리를 해주었습니다. 상태에 따라 렌더링 방식을 바꾸어 주었고, 상태는 총 6가지로 Intro, Start, Shop, Help, Destroy, Game Over로 나누었습니다.  **\* 프로젝트 진행 시 있었던 문제 해결 및 진행 과정**  - 처음 작업하기 전에 우선적으로 클래스 설계를 어떻게 하면 좋게 할 수 있을지를 고민을 많이 했습니다. 상태에 따라 변경할 수 있도록 설계를 했었고 프로그래밍에 대해 아직 미숙한 부분이 많았을 때여서 메시지 값을 어떤 형식으로 받아와서 처리 해야 할지를 몰라서 잘하는 친구와 Google 검색을 통해 많은 정보를 얻었습니다. 난이도를 추가 했고, 일정 거리마다 적의 체력을 증가 시키는 방법. 또한, 코인을 모아서 상점에서 무기를 강화 할 수 있고, 듀얼샷 아이템을 추가해 일정시간 동안 무기가 두발씩 나가도록 만들었습니다. 충돌처리 부분이 가장 힘들었는데, 수치 값이 정확하지 않아 생긴 일이라는 것을 알아서 그림을 그리면서 정확히 맞추었습니다. 또, 객체가 선언이 안될 때가 있었는데, 그 부분은 헤더가 중복으로 호출로 인해 생겨 중복을 최소화 시켜서 해결을 했습니다.  **\* 프로젝트를 마무리 하면서 느낀 점**  - 확실시 콘솔 창으로 제작하는 것 보다 그림이 보이는 게 있어서 재미있었고, 또, 헤더 중복 선언을 최소화 해야 하는 것을 느꼈습니다. 이 기말 과제에서 1위를 해서 몹시 기뻤습니다.  **\* 유튜브 링크 :** [**https://youtu.be/dXZrARfQd4Q**](https://youtu.be/dXZrARfQd4Q) | | |

|  |
| --- |
| **<클래스 다이어그램>**  모든 오브젝트는 Game Object에서 상속받아 사용이 되고, Game State에서 모든 상태를 상속을 받아서 재정의로 만들었습니다. My Image에서는 모든 리소스를 받아 와서 불러와서 사용 할 수 있게 하였고, Objet List에서는 Linked List를 만들어서 무기를 관리를 해주었습니다.    **<Linked List>**  수업 시간에 배운 리스트를 사용해서 플레이어의 무기를 동적으로 생성하고 삭제할 수 있도록 만들었습니다. Insert Object에서 생성을 해주고, Remove Object에서 삭제를 해주고 있습니다. 이 개념을 배우고 이 프로젝트에 사용 하면 편리 할거 같았고, 잊어버리기 전에 복습하는 생각으로 사용을 했습니다. |