GAME PROGRAMMER

KWON OH HYUN

INFORMATION

PROFILE

이름 : 권오현
생년월일 : 1994. 07. 19
주소 : 수원시 권선구 오목천동 한화 꿈에 그린 아파트 216동 1004호
E-mail : ohhyun5442@gmail.com
휴대전화 : 010.2719.5330

EDUCATIONAL BACKGROUND

2011. 03 ~ 2013. 02 / 유신 고등학교 졸업
2013. 03 ~ 2019. 04 / 청주 대학교 컴퓨터정보공학과 중퇴

LICENSE

2013. 02 1종 운전면허증 취득
2015. 07 정보처리기능사 자격증 취득

ACTIVITIES

2017. 03 ~ 2018. 03 / Jusin 게임 아카데미 수료
2018. 07 ~ 2020. 03 / 공게임즈 스크린야구 메인 개발
2020. 03 ~ / 공게임즈 이사만루3 클라이언트 콘텐츠 개발

MILITARY

2014. 06 ~ 2016. 03 / 대한민국 육군 병장 전역

GAME PROGRAMMER

KWON OH HYUN

ABILITY

USING ABILITY

* C / C++ / C#
  → 절차 지향과 객체 지향적 언어의 이해와 포인터와 함수 등을 이용하여 게임을 개발하였습니다.

* Unity
  → 유니티를 사용한 UI, 컨텐츠를 개발을 하였으며 커스텀 에디터를 통하여 툴 제작 및 스크립트를 개발하였습니다.

* AOS / IOS
  → 안드로이드 스튜디오 및 XCode를 통한 디버깅 및 플랫폼 빌드를 하였습니다.

* STL(표준 템플릿 라이브러리)
  → vector : 인벤토리를 구현할 때 아이템들을 보관할 때 vector를 이용하여 보관을 하였습니다.
  → list : 오브젝트들을 보관할 때는 List에 보관하여 이터레이터 패턴을 이용하여 랜더를 하였습니다.
  → map : 매디에이터 패턴을 사용하여 오브젝트들을 보관할 때 Key값을 이용하여 오브젝트들을 탐색할 수 있게 구현하였습니다.

* WINAPI
  → 기본적으로 WinApi에서 제공하는 TransparentBlt을 통하여 DC에 Sprite 이미지를 그려 움직이는 것처럼 구현하고 GetAsyncKeyState함수를 통하여 키 입력을 체크할 수 있게 구현하였습니다. 더블버퍼링을 이용하여 화면을 출력하고 다음 출력할 것을 미리 그려 덮어 씌우는 형식으로 화면의 깜빡임을 해결하였습니다.

* MFC
  → MFC 계층구조에 대한 이해와 View클래스와 Form클래스를 이용하고 트리 컨트롤, 슬라이드 컨트롤 등을 이용하여 여러가지를 이용한 Tool 제작하였습니다.

* DirectSDX
  → Graphic Device를 초기화하거나 Direct내부에 존재하는 함수와 구조체를 이용한 게임제작을 제작하였습니다.
  → 렌더링 파이프라인에 대한 이해와 뷰 포트에서 로컬 좌표로 변화시켜 지형 피킹을 하거나 UI피킹을 하였습니다.
  → 더블 버퍼링에서 스왑 체인으로 바꿔 렌더링을 하였습니다.

GAME PROGRAMMING

KWON OH HYUN

ABILITY

USING ABILITY

* 자료구조

→ 큐 : 후입 선출(Last In First Out)의 이론의 이용하여 먼저 들어간 것이 나중에 Pop되도록 구현하였습니다.
→ 스택 : 선입 선출 (First In First Out)의 이론을 이용하여 먼저 들어간 것이 먼저 Pop되도록 구현하였습니다.
→ 싱글 링크드 리스트 : 성적표에서 정보들을 노드들을 이용하여 서로 연결시켜 보관하였습니다.
→ 이진 트리 : 링크드 리스트를 이용하여 만든 트리 구조에서 데이터를 코스트에 따라 탐색하도록 구현하였습니다.

* 디자인 패턴

→ 싱글톤 패턴 : 객체들을 생성을 할 때 단 한번만 생성하도록 제한을 하기 위하여 사용하였습니다.
→ 스테이트 패턴 : 객체들의 상태별로 관리하여 움직임을 제어하였습니다.
→ 컴포넌트 패턴 : 필요한 기능들을 Component화 시켜 만들고 객체들에게 복제하여 사용하도록 하였습니다.
→ 이터레이터 패턴 : List에 객체들을 담아 놓고 이 객체들의 Update와 Render함수를 이터레이터를 돌아서 수행하도록 구현하였습니다.
→ 메디에이터 패턴 : 오브젝트들을 오브젝트 매니저로 중계자의 역할을 하여 보관하도록 하였습니다.
→ 스트래티지 패턴 : 공통되는 객체들을 한 개의 부로객체로 묶어서 상속받아 쓰도록 구현하였습니다.
→ 프로토타입 패턴 : 필요한 객체들을 미리 생성하여 필요할 때 생성한 객체를 복사해서 사용 하도록 하였습니다.
→ 팩토리 패턴 : 여러 객체들이 사용해야 하는 함수를 사용해야하는 경우 함수를 템플릿화 시켜서 함수를 호출하도록 구현하였습니다.
→ MVC 패턴 : 기본적인 씬 구조를 구현할 때 사용을 했으며 MODEL, VIEW, CONTROLLER를 기준으로 각각의 역활에 맞게 구현했습니다.

* 수학

→ 내적 : 몬스터의 위치를 탐색하거나 네비게이션의 내부에 있는지 없는지 판별하는 것을 구현하였습니다.
→ 외적 : 가고 싶은 방향을 정해주거나 Frustum Culling을 사용할 때 내부에 있는지 없는지 판별하기도 하였습니다

* 물리

→ 중력 가속도 : 점프를 하면 떨어지도록 구현을 하였습니다.
→ 포물선의 법칙 : 총알의 날라가면 떨어지는 것을 구현하였습니다.
→ 등속도의 운동 : 일정한 속도로 객체가 움직이도록 구현하였습니다.