|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **작성자** | **2017180012 박주용**  **2017180029 이예준**  **2017182030 이수민** | **팀명** | FruitsPangPang |
| **주차** |  | **기간** | **2022.03.16~2022.03.22** | **지도교수** | **정 내 훈**(서명) |
| **이번주 한일** | * **박주용**   **LGT 기능 추가, 애니메이션 임포트 및 수정, 맵 landscape 수정**   * **이예준**   **애니메이션 캐릭터에 연결, 근접 과일 스폰**   * **이수민**   **AIClient와 FPPClient 통합, itemcode 추가** | | | | |

**<상세 수행내용>**

* + **박주용**

지도이(가) 표시된 사진

자동 생성된 설명

* LGT(Landscape Grass Type)를 통한 잔디, 나뭇가지, 자갈 깔기 자동화.

1. LGT\_Grass: Landscape grass layer의 데이터를 가져와서 맵의 grass texture 위에만 잔디가 생성되게 하였다. Lerp를 통해 물(puddle) 웅덩이와 cliff texture에는 잔디가 생성되지 않게 하였다. 잔디의 종류는 8가지를 다양하게 섞었고 퍼포먼스를 위해 5000거리 까지만 잔디가 보이게 함.
2. LGT\_Sand: 잔디와 같은 방법으로 구현했고 sand 데이터를 가져와 sand위에 5종류의 작은 나뭇가지와 자갈들을 적절히 섞었다.

텍스트, 잔디, 실외이(가) 표시된 사진

자동 생성된 설명

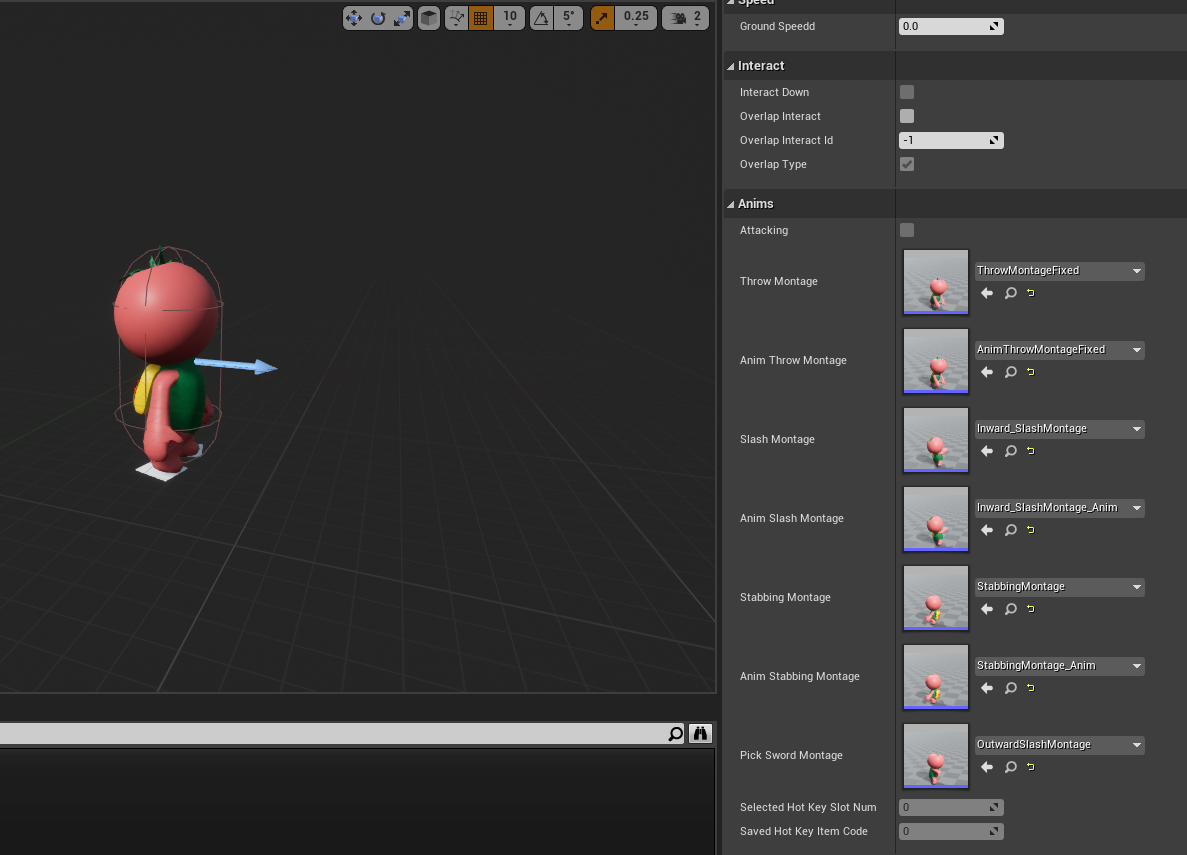
<맵 바닥 캡처 화면>

텍스트, 스크린샷, 모니터, 실내이(가) 표시된 사진

자동 생성된 설명

애니메이션((Inward/Outward)slash, stabbing, climbing, death, hit) 임포트, 키 프레임 수정

* + **이예준**



칼 휘두르고 찌르는 애니메이션 동작되게 만듦.

텍스트, 잔디이(가) 표시된 사진

자동 생성된 설명

근접 무기 슬롯을 선택했을 때 근접 무기(당근, 대파)가 스폰

* + **이수민**

//AIClient & FPP 두개의 클라이언트를 합치는 작업에 있어서 **첫 번째** 방향성

- 맵 수정 또는 기능 구현에 있어서 두번 작업을 하는 것은 너무 번거롭기에 합치기로 결정.

- 기본적으로 FPP의 Network class는 AIClient의 N:1 통신 방식을 따라갈 것.

- FPP는 기존 1:1 통신이었지만, N:1로 했을 때 N-1개의 포인터가 낭비되지만 이것보다 편의성으로 얻는 이점이 더 크기 때문에 신경 쓰지 않기로 한다.

- Network class를 두개를 각각 분리한다면? ex) Network.h, Network.cpp / AiNetwork.h, Network.cpp

장점: 배포할 때 Ai Network만 빼면 보안 신경 쓸 필요가 없다.

단점: 패킷을 보내야 할 때, 여러 .cpp에서 Cast를 해서 AI socket을 쓸건지 Player socket으로 쓸건지 구분해야 한다.

- Network class를 하나로 묶는다면?

장점: Network.cpp 에서만 Cast를 통해 AI인지 Player인지 구분하면 된다. 여러 .cpp파일에서 쓰는건 유지보수적으로 좀 힘들다고 판단됨.

-> 이유: 실 코딩을 하게 된다면 AI와Player가 둘 다 사용될 수 있는 Object ex)Tree, Punnet같은 경우에서는 Tree.cpp에서 AI Cast 시도하고 false면 Player Network class 사용, Punnet.cpp에서 AI Cast시도하고 false면 Player Network class 사용, 이런 식으로 코딩을 할텐데, 여러 곳에 흩뿌려져 있기보다는 Network.cpp에 몰아넣어 놓는게 나을 것 같다는 판단.

-> 물론 call되는 빈도에 따라 성능차이가 있을 수도 있겠지만, 드라마틱한 성능향상이 아닐 것 같아서 생각하지 않기로 한다.

단점: 배포할 때 합쳐져 있는 code를 골라서 빼는게 귀찮은 점?

이러한 이유로 모듈을 합치는 걸로 결정.

- AIClient의 방식으로 바꾸면서 해야 할 것.

1) process\_packet2의 구조를 따올 것.

2) AICharacter와 MyCharacter는 기본적으로 Character를 상속받으므로

Network에서 기존 My로 관리하던 것을 Character로 받은 다음, Cast해서 사용하기로 한다.

//3일정도 더 생각한 다음 결정된 **두 번째** 방향성

이전 커밋에서 Network class를 합친다고 했는데,

생각해보니 굳이 Network // AiNetwork 이런 식으로 분리를 안해도

overlapped recvCallback만 구분해주면 알아서 분리가 되고, 이 방식이 가장 간단하면서 명료하기 때문에 Network class는 합치되, Callback을 Ai전용, Player전용을 각 따로 두는 방식을 채택.

후에 배포용으로 Ai Logic을 뺀다고 했을 때도, callback func부분만 지워주면 되니 수월하게 작업이 가능하다.

또한 AICharacter, MyCharacter가 상속받는 “BaseCharacter”를 새롭게 두어 둘이 동시에 사용하는 것 (ex) Network 관련, Inventory 등등)은 BaseCharacter에 넣어서 첫 번째 방향성의 단점(여러 .cpp에서 Cast를 해서 AI인지 Player인지 구분해야 한다는 점)을 상속으로 해결했다.(BaseCharacter의 socket을사용하면 알아서 해결)

버그 수정

- 부활 시 클라가 터지는 현상 수정 (Ai Client의 EndPlay를 넣어 줌으로 해결)

각 ItemCode 추가.

|  |  |  |  |
| --- | --- | --- | --- |
| **문제점 정리** |  | **해결 방안** |  |
| **다음 주차** |  | **다음 기간** | **2022.03.23~2022.03.29** |
| **다음주 할 일** |  | | |
| **지도교수**  **Comment** |  | | |