

[Toy Ground(토이 그라운드)]

| | | | |
|-------------|--|-----|-----|
| 19 주 | 2020. 5. 2 ~ 2021. 5. 8 | 작성자 | 김영준 |
| 이번 주 한 일 | <p>[0] 공동 05.05 주간 회의 - 일주일간 계획공유 - 교수님 면담 피드백 수정 - 패킷 관련 회의</p> <p>[1] 김영준(서버) 주간목표: - Client - Server 연동(이동 제외한 로직 제외) - Map 로드, Bullet 구현, 게임 종료 패킷 구현</p> <p>05.02 ~ 04. 교수님 검토 전 데모 게임 구현, 교수님 검토 패킷을 주고 받는 과정에서의 버그는 더미 클라이언트를 통해 확인하고 수정, 더미 클라이언트 ~ 서버 연동이 문제없이 돌아가는 것을 확인하고 실제 클라이언트에 network 클래스를 추가하여 클라이언트에서 접속하고 패킷을 교환할 수 있도록 구현함.</p> <p>졸업한 선배의 특강을 들었을 때 모든 클라에 네트워크를 연결할 때 통용할 수 있는 본인만의 netCore을 만든 모습을 봤는데 일정에 급급하여 허술한 설계로 구현한 것 같다.</p> | | |

```

enum SERVER_TYPE { ST_LOBBY = 0, ST_BATTLE = 1, ST_COUNT = 2 };

enum EVENT_TYPE { EV_RECV = 0, EV_SEND = 1 };

+struct EXOVER { ... };

+struct ConnectSocket { ... };

+struct CLIENT { ... };

class Network : public TemplateSingleton<Network> {
public:
    Network();
    ~Network();

    void DoWorker();
    void ProcessPacket(int id, unsigned char packet[]);

    void ConnectLobbyServer();
    void ConnectBattleServer(int id);

    void DisconnectClient(int id);

    void SendLoginPacket(int id);
    void SendRequestUserInfo(int id);
    void SendUpdateUserInfo(int id, int mmr);
    void SendAutoMatchPacket(int id);

    void SendBattleLoginPacket(int id);
    void SendJoinPacket(int id, int room_no);
    void SendReadyPacket(int id);
    void SendGameStartPacket(int id);

    void SendKeyDownW(int id);
    void SendKeyDownA(int id);
    void SendKeyDownS(int id);
    void SendKeyDownD(int id);

    void SendKeyUpW(int id);
    void SendKeyUpA(int id);
    void SendKeyUpS(int id);
    void SendKeyUpD(int id);

    void SendPacket(int id, void* packet, SERVER_TYPE st);

    XMFLLOAT3 GetPos() const;

    void error_display(const char* msg, int err_no);

```

05.05 교수님 미팅 이후 회의 및 작업일지 최신화

교수님과의 미팅 이후 중간 발표까지 완성할 내용을 수정했다. 게임의 생명주기를 한번 돌릴 수 있는 모습을 보여주기 위한 우선의 과제를 선정하여 회의를 통해 순서를 정하고 일정을 수립했다.

또한 클라와 회의를 통해 구현 업무에 대한 분업을 문서화 하여 클라와 서버간 연동을 하면서 정해야 할 내용, 앞으로 정해야 할 내용을 정함

구현 업무 확실히 나누기

1. 이동 키 입력

1. C에서 키입력
2. C → S 키 입력 패킷 (key down, key up + key val(WASD+Jump+Bullet) + look vector)
3. S에서 키 입력에 따른 좌표 변경
4. S → C 좌표 변경 패킷 전송 (id, position vector, look vector)

2. coin 생성 / 획득 / 잃어버림

1. 생성

1. (S) 특정위치, 특정시간에 coin 생성, 특정조건(사망)에 coin 생성
2. S → C coin 생성 패킷 (coin id, position vector)

2. 획득

1. (S)C 이동으로 Toy와 coin의 충돌 발생
2. S → C coin 획득 패킷(id, 사라진 coin id)

3. 잃어버림

- + :: 1. (S) Bullet 이동으로 Toy 사망
- 2. S → C Toy 사망 패킷(id, 사망 패널티 시간, 사망 position vector), coin 생성

3. 총알

:: 1. 생성

- + :: 1. C→S 키 입력 패킷(키 입력에서 처리) (look vector, bullet id)

2. 이동, 충돌

1. (S) 매 프레임 bullet 좌표 갱신 및 충돌 검사
2. 충돌 처리 (사라지거나, Toy에게 영향을 주거나, obj에 영향을 주거나)
3. S→C 갱신된 좌표 전송(bullet id, position vector)
 1. S→C 혹은 갱신된 상태 전송(bullet id, disable)

05.06 Map class 구현

회의를 통해 게임 로직을 처리하는 서버에서 map의 정보를 가지고 클라이언트에게 넘겨주는 방식을 사용하기로 함, 기존의 클라이언트에서 맵을 로드하던 코드를 수정하여 맵 정보를 가지고 충돌처리 등 게임 로직을 처리할 수 있도록 구현할 예정

```

void Map::LoadMapInfo(string mapName) {
    string path = "Maps\\" + mapName + ".txt";
    cout << path << endl;
    std::ifstream fileIn(path);

    if (fileIn) {
        int ri, rj;
        for (int k = 0; k < MAP_HEIGHT_BLOCK_NUM; ++k) {
            int floor;
            fileIn >> floor;
            for (int i = 0; i < MAP_DEPTH_BLOCK_NUM / 2 + 1; ++i) {
                for (int j = 0; j < MAP_WIDTH_BLOCK_NUM; ++j) {
                    // 배열에 맵 저장
                    int input;
                    fileIn >> input;
                    ri = MAP_DEPTH_BLOCK_NUM - i - 1;
                    rj = MAP_WIDTH_BLOCK_NUM - j - 1;
                    data[k][i][j] = input;
                    data[k][ri][rj] = input;
                }
            }
        }

        for (int k = 0; k < MAP_HEIGHT_BLOCK_NUM; ++k) {
            for (int i = 0; i < MAP_DEPTH_BLOCK_NUM; ++i) {
                for (int j = 0; j < MAP_WIDTH_BLOCK_NUM; ++j) {
                    cout << data[k][i][j];
                }
            }
            cout << endl;
        }
        cout << endl;
    }
}

```

05.07 netCore 구현 시작

기존의 network에 작업을 하지 않고 새로 클라이언트 연결 전용 클래스를 제작하여 구현하려 함, 클라이언트 연동 관련된 예제가 부족해 git 검색을 05.04부터 하고 있었고 원성현 게임공학과 선배님의 2년전 코드 구조를 참고하여 구현하려고 함(io와 event 처리를 따로 나누는 방식) 주말 이전에 제작 완료하여 화요일 교수님 미팅 전 클라이언트 연동을 netCore을 통해 할 것

[2] 김동석(클라이언트)

주간 목표 :

- FBX Converter 수정
- Cowboy 캐릭터 추가

05.02 버그 수정

- FBX Converter 수정

->문제점

FBX 파일을 FBX Converter에서 제대로 읽어오지 못했다.

->원인

Root Node의 Child Node에서 정보를 받아오도록 했었는데 예제 Coverter를 사용해 확인해본 결과 CowboyAsset의 Root Node 바로 밑의 Child Node에는 쓰레기 값이 들어있었다.

->해결 방안

Root Node의 두 단계 밑의 Child Node에서부터 파일의 정보를 읽어 오도록 수정했다.

- Scene 전환 시 카메라 모드가 Free Mode가 되는 오류

->문제점

Lobby Scene에서 Gameplay Scene으로 'T'를 눌러 Scene 전환 시 카메라 모드가 Third Person Mode가 아닌 Free Mode로 전환된다.

->원인

GamePlay Scene에서 Free Mode로의 카메라 전환키가 'T'여서 Scene전환과 동시에 Free Mode가 되는 것이다.

->해결 방안

GamePlay Scene의 Free Mode 전환키 'F3'으로 수정

- Cowboy 에셋 Scale 문제

->문제점

저번 주에 Unity에서 Scale을 수정해 Cowboy 에셋의 크기를 줄였는데 이게 Unity에서만 적용되고 ToyGround에서는 원래 에셋의 크기로 나온다.

->Unity의 Skinned Model 관련해서 공부를 해봐야겠다.

05.03 클라이언트와 서버 연결(교수님 면담 전 최종 점검)

- LobbyScene 수정

서버로부터 정보를 받아오도록 수정

- PlayerController 수정

Character 이동 관련 서버에서 관리하도록 영준이와 상의 후 수정

05.04 윤정현 교수님과의 면담

- **면담 후 피드백 관련 회의**

발표준비를 어떻게 할지 회의, 교수님 피드백 정리

05.05 주간 회의

- **구현 업무 확실히 나누기**

이동, 보석 생성, 총알 등 Client와 Server의 작업을 개인, 공통으로 분류하고 패킷 관련해서 마지막으로 정했다.

- **일주일간 일정 계획**

```
C (이동, 총알, 총돌, bb, 애니메이션)
~금 오후 (bb/애니메이션)
~토 야간 (총알)
~일 야간 (이동, 총돌)

S(이동, 총알, 총돌, coin, network, 맵 로드)
~금 오후 (coin, network, 맵 로드)
~토 야간 (총알)
~일 야간 (이동, 총돌)

화: pt제작
```

[일주일 계획]

- **Cowboy 에셋 애니메이션 관련 오류 원인 찾음**

.skeleton, .mesh파일은 문제가 없으나 .Anim파일의 keyframe의 수가 부족하다.

- **MapLoad 서버로 이동**

영준이와 회의를 통해 Client에서하고 있던 MapLoad를 서버로 옮겼다.

앞으로 프로젝트에서 Server와 Client가 동시에 작업할 수 있도록 Define을 사용해 분리하는 작업을 할 것이다.

05.07 Cowboy Animation

- **오류 수정을 위해서 수많은 시도**

->원인

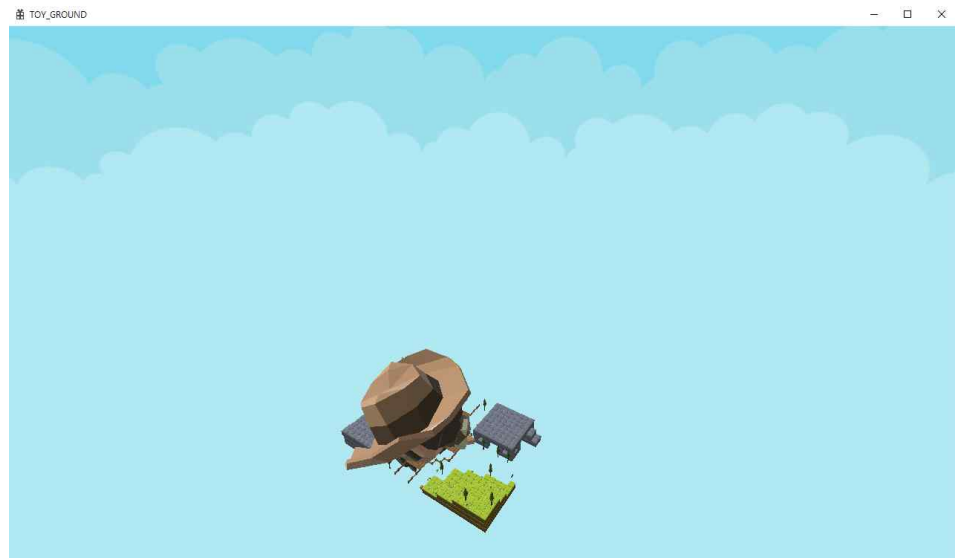
근본적인 원인은 지금 구현해 놓은 Test Asset과 다르게 우리가 구매한 Cowboy 에셋은 에셋 내에서 애니메이션을 제공하지 않는다. 그래서 다른 무료 애니메이션 에셋을 Cowboy에셋에 붙여서 사용해야 하는

데 Cowboy 에셋의 계층구조가 다른 무료 에셋들에 비해 특이하다.

->시도

1. Cowboy 에셋에 맞는 무료 애니메이션 에셋 찾기

12개의 무료 애니메이션 에셋들을 받아서 적용해 봤지만 Cowboy의 계층구조와 맞는 에셋이 없어 Fbx 파일로 변경 후 프로젝트의 적용 시 기괴하게 움직인다.



[텍스처와 모델은 제대로 입혀졌지만 기괴하다]

2. 직접 애니메이션 제작

Unity에서 제공하는 Animation Controller를 사용해서 직접 Idle 애니메이션을 제작했다.

->문제점1

Unity에서는 자연스럽게 애니메이션이 동작하지만 KeyFrame을 3개로 설정했는데 FBX로 전환 시 유니티에서 제공하는 자동 보간이 모두 KeyFrame으로 들어가 파일이 엄청나게 커진다(거의 1000개 가까이 되는 KeyFrame)

->문제점2

Idle 애니메이션은 어떻게든 만든다고 해도 Runnig 이나 Jump 같은 애니메이션은 만들 엄두가 안난다.

3. Cowboy 에셋의 계층구조를 보편적으로 수정한다.

Cowboy 에셋의 Prefeb과 Avatar를 수정하려고 여러 방법을 시도해 봤지만 실패...

4. 다른 에셋을 찾아볼까..

하루종일 찾아봤지만 방법이 없어 다른 에셋을 찾아볼까 생각도 했지만 기획할 때부터 ToyGround에 가장 적합하다고 생각해 구입한 에셋이기 때문에 버릴 수 없었다.

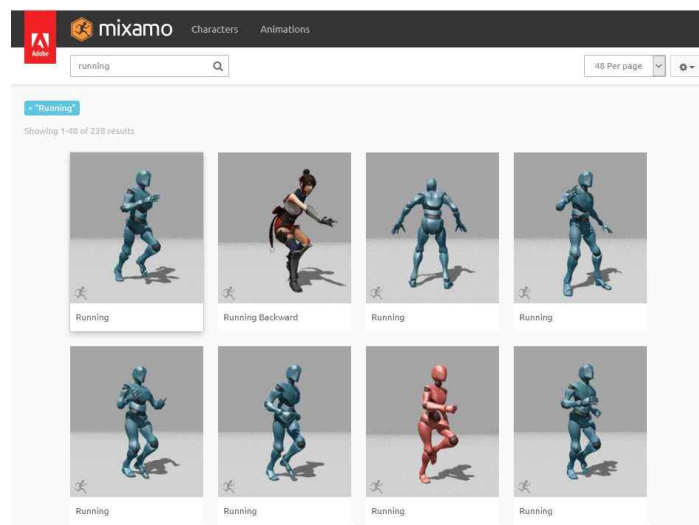
이 에셋을 구매한 사람들은 어떻게 애니메이션을 적용했을까 확인해보려고 리뷰를 보러갔는데 다들 나와 같은 이유로 슬퍼하고 있는 걸 읽다 보니 에셋을 제작한 회사의 Youtube 채널이 있다는 리뷰를 발견해 바로 찾았다.

->해결 방안


에셋 제작 회사의 Youtube 채널에 애니메이션을 적용하는 방법이 있었다... 에셋을 판매할 때 링크라도 달아줬다면 얼마나 좋았을까...

->결론

아래의 사이트에서 애니메이션을 받아와 적용했다.



[영상에 나온 사이트 www.mixamo.com]

| | |
|---------------------|---|
| | <div data-bbox="392 273 1350 828">  </div> <div data-bbox="625 842 1114 880"> <p>[Cowboy 애니메이션을 적용한 모습]</p> </div> <div data-bbox="392 936 730 974"> <p>05.08 Animation 마무리</p> </div> <div data-bbox="392 981 699 1019"> <p>- Cowboy 에셋 Scale</p> </div> <div data-bbox="392 1025 1289 1064"> <p>Cowboy 에셋의 크기를 ToyGround의 월드 크기에 맞춰 조정했다.</p> </div> <div data-bbox="392 1122 678 1160"> <p>- Attack Animation</p> </div> <div data-bbox="392 1167 1350 1301"> <p>총을 쏘는 애니메이션은 많지만 Cowboy 캐릭터처럼 양손에 권총을 들고 쏘는 애니메이션은 없었다. 따라서 Punch Combo라는 두 주먹을 번갈아가면서 내지르는 무료 애니메이션을 수정해 사용할 예정이다.</p> </div> |
| <p>다음 주 할 일</p> | <div data-bbox="392 1301 501 1339"> <p>[0] 공동</p> </div> <div data-bbox="392 1346 735 1384"> <p>05.11 주간 회의 (20주차)</p> </div> <div data-bbox="392 1391 676 1429"> <p>안건 : 일일계획 공유</p> </div> <div data-bbox="392 1435 722 1473"> <p>교수님 면담 피드백 수정</p> </div> <div data-bbox="392 1532 622 1570"> <p>[1] 김영준 (서버)</p> </div> <div data-bbox="392 1576 509 1615"> <p>주간목표</p> </div> <div data-bbox="392 1621 802 1756"> <ul style="list-style-type: none"> - netCore 완성 - coin, bullet 관련 패킷 완성 - 중간발표 이전 점검 </div> <div data-bbox="392 1812 708 1850"> <p>[2] 김동석 (클라이언트)</p> </div> <div data-bbox="392 1856 525 1895"> <p>주간 목표</p> </div> <div data-bbox="392 1901 967 1986"> <ul style="list-style-type: none"> - 서버와 클라이언트(이동, 충돌) 공동 작업 - 간단한 UI를 위한 빌보드 </div> |

| | |
|-----|--|
| | <ul style="list-style-type: none"> - 충돌처리, BB 구현(서버 내에서) - 중간발표 전 최종 점검(버그 수정 등) |
| 문제점 | <p>[1] 김영준 (서버)</p> <ul style="list-style-type: none"> - netCore와 같은 구조를 한번에 만들지 않고 급한 마음에 만든 network 클래스의 구조를 다시 보니 정말 비효율적인 코드였다, 지금까지 만든 서버 코드는 다른 예제가 많았지만, 클라이언트 연동을 위한 코드는 많이 접하지 못해 어려움을 겪었지만, git의 다른 많은 예제를 보고 새로운 방향성을 찾을 수 있었다. - 중간발표가 2주도 채 남지 않았다, 반드시 일정 내에 게임의 시작부터 끝까지 구동할 수 있도록 노력하겠습니다. <p>[2] 김동석 (클라이언트)</p> <ul style="list-style-type: none"> - Cowboy 에셋에 애니메이션을 적용하느라 엄청나게 고민한 주였다. 앞으로 에셋을 선택할 때 애니메이션 포함 여부를 확인해야겠다. - 저번 주부터 영준이와 클라이언트 서버 연동작업을 하는데 중간발표 전에 계획대로 마무리를 잘 해야겠다. |

[추가]