**네트워크 게임 프로그래밍**

**Term Project**

**최종 보고서**

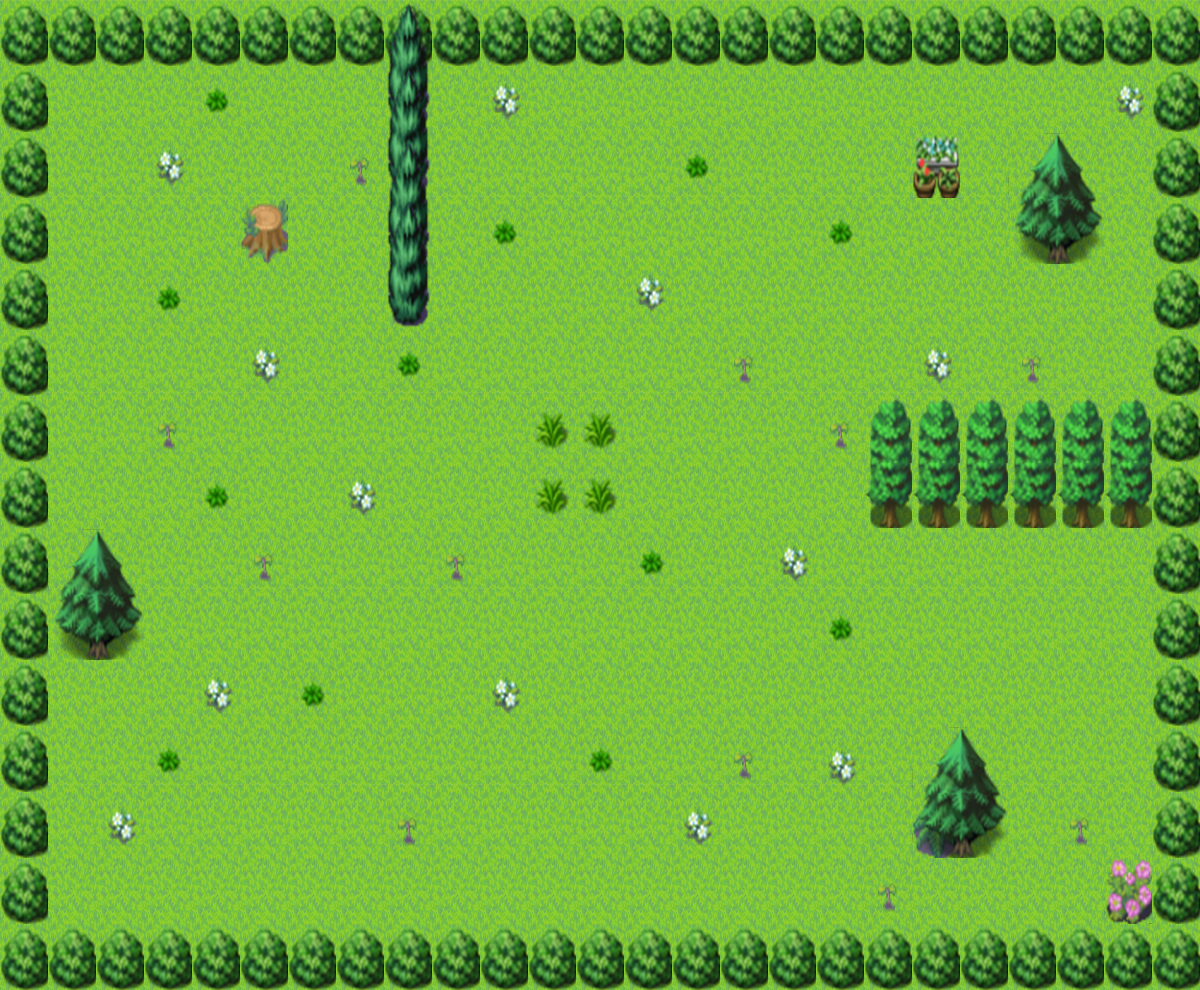
**2012180016 배강산**

**게임 기획**

* 1. **게임설명**
* 1 VS 1 ~~VS 1~~[[1]](#footnote-1) 구도의 2D 슈팅대전게임을 TCP/IP 프로토콜을 사용하여 개발. 특성이 다른 세 종류의 캐릭터 중 하나를 선택해 최후 1인까지 살아 있으면 승리
* 각각의 캐릭터는 그 캐릭터에 해당하는 발사체를 발사할 수 있다.
* 발사체는 직선경로를 따라 이동한다.
* 발사체가 장애물이나 다른 캐릭터를 만나면 데미지를 주고 없어진다.
  1. **게임 조작법**
* 방향키: 키보드 W, A, S, D 키
* 에임 조준: 마우스 위치
* 발사: 마우스 좌 클릭
* 게임 윈도우 사이즈 1200 X 990 pixel ( 25 X 15 칸 )
  1. **게임 캐릭터 ( 48 X 66 pixel )**

|  |  |  |  |
| --- | --- | --- | --- |
| **캐릭터** | **특징** | **이미지** | **발사체(24X24)** |
| **거너** | 총알을 쏜다 연사 속도: 중 파괴력: 중 |  |  |
| **바머** | 폭탄을 던진다 연사 속도: 하 파괴력: 상 |  |  |
| **마법사** | 마법을 쓴다 연사 속도: 상 파괴력: 하 |  |  |

* 1. **게임 맵**

****

**※ 맵 사이즈 1200 x 990 pixel (윈도우 사이즈와 동일하게)  
- 플레이어 순서대로 지정된 위치에 뿌려준다  
- 이동할 수 없는 장애물 구성**

* 1. **클라이언트와 서버 구현  
     Server  
     <접속>**
* 클라이언트가 접속하면 스레드를 하나 만들어 해당 클라이언트에 부여한다.
* ~~4~~ 3[[2]](#footnote-2)개 이상의 클라이언트에 대한 접속을 거부한다.
* 접속한 클라이언트들의 준비 상태를 확인한다.
* 모든 클라이언트들이 준비 되면 3초 후 게임을 시작한다.

**<진행>**

* 클라이언트로 캐릭터 정보를 보낸다.
* 클라이언트로부터 ~~캐릭터 상태 갱신~~ [플레이어 상태 패킷][[3]](#footnote-3)이 오면 캐릭터의 상태를 갱신해 준다.
* ~~클라이언트로부터 발사체 발사 시도 패킷이 오면 발사 가능한 상태인지 확인한다.~~
  + ~~발사 가능한 상태라면 발사체를 생성해준다.~~[[4]](#footnote-4)
* 각각의 캐릭터의 상태 정보에 의해 각 캐릭터의 위치를 갱신한다.
* 발사체 위치를 갱신하고 캐릭터의 발사 시도 상태도 같이 확인하여 발사체를 생성한다.[[5]](#footnote-5)
* ~~캐릭터 위치를 갱신할 때, 캐릭터가 가지고 있는 발사체들의 위치를 갱신한다.~~
* 캐릭터들과 장애물들과 발사체들 간의 충돌을 검사한다.
* 충돌이 이루질 경우 어떤 타입을 가진 클래스들의 충돌인지 확인한다.
  + 캐릭터 – 장애물 : 장애물을 뚫지 않도록 처리한다.
  + 캐릭터 – 캐릭터 : 서로 뚫고 지나가지 않도록 처리한다.
  + 캐릭터 – 발사체 : 캐릭터에 데미지 처리를 해주고 발사체를 없앤다.
  + ~~발사체 – 장애물 : 장애물을 뚫지 않도록 처리한다.~~[[6]](#footnote-6)
  + 발사체 – 발사체 : 처리하지 않는다.
* 캐릭터 생명력이 0이하로 내려가 죽은 캐릭터가 생겼을 경우 처리를 해준다.
* 접속한 모든 클라이언트들에게 모든 캐릭터에 대한 정보를 보낸다.

**Client**

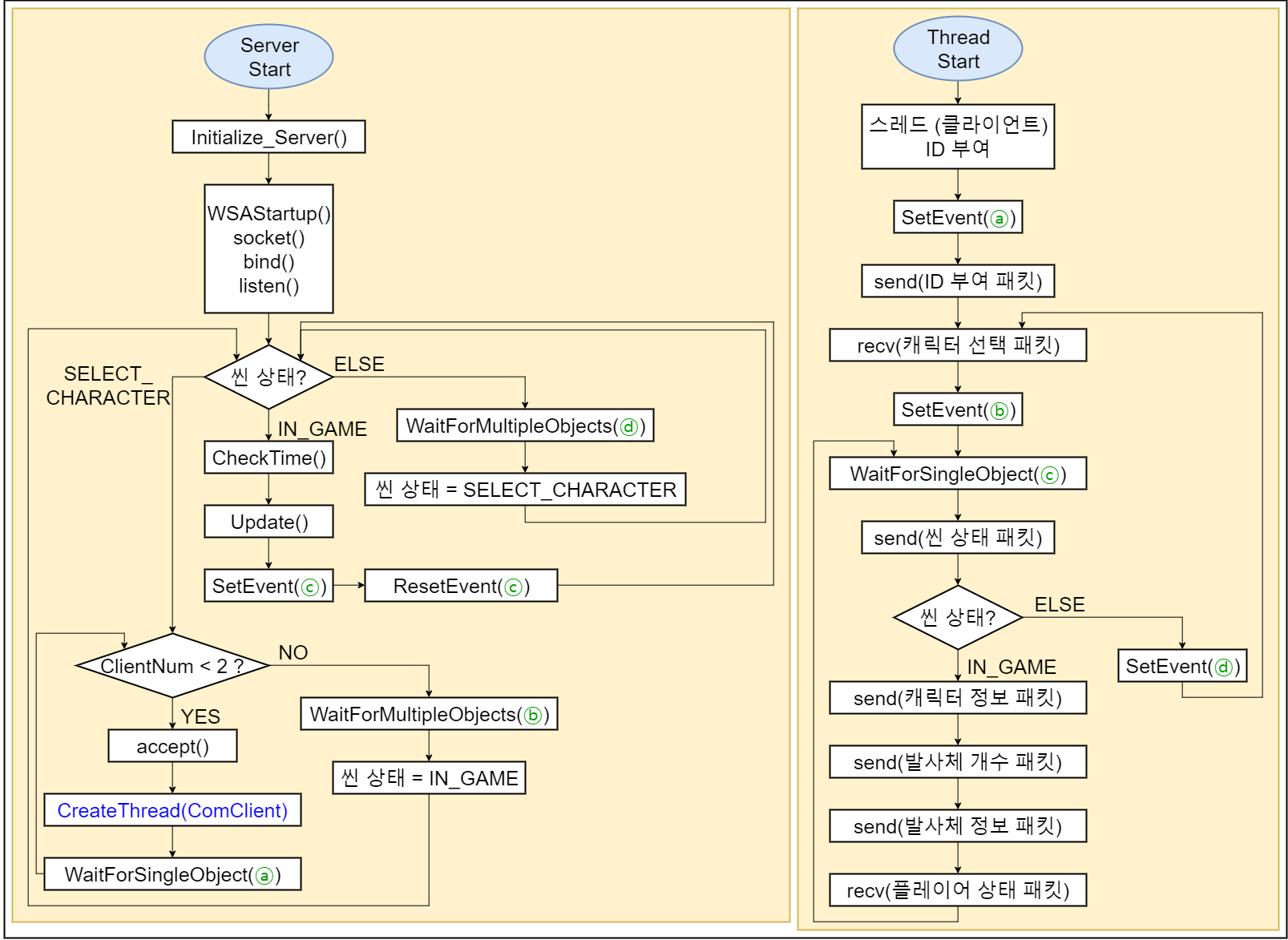
**<접속>**

* 플레이어가 준비를 하면 캐릭터 선택 정보를 서버로 보낸다.

**<진행>**

* 키보드 입력에 따라 자신의 상태를 결정한다.
* 자신의 상태 정보를 서버로 보낸다.
* ~~플레이어가 발사 시도를 하면 서버로 발사 시도 패킷을 보낸다.~~[[7]](#footnote-7)
* 서버로부터 받은 정보를 이용해 자신이 가지고 있는 캐릭터 배열, 발사체 배열[[8]](#footnote-8)을 갱신한다.
* 배경을 그린다.
* 캐릭터들을 그린다.
* ~~캐릭터를 그릴 때, 캐릭터가 가지고 있는 발사체 벡터를 이용해 발사체를 그린다.~~
* 발사체들을 그린다.[[9]](#footnote-9)

1. **High-level Design**
   1. **Server**
      * **멀티 스레드를 사용하여 클라이언트를 ~~3~~ 2[[10]](#footnote-10)개까지 지원한다. 메인 스레드가 있고 클라이언트 하나에 스레드를 하나씩 추가한다. (총 스레드 개수 : ~~4~~ 3[[11]](#footnote-11)개)**
      * **메인 스레드 - 클라이언트 하나가 접속하면 ‘클라이언트 통신 스레드’를 생성하고 소켓을 할당한다. 이벤트를 사용하여 생성한 스레드가 ID를 부여 받기를 기다렸다가 새로운 클라이언트 접속을 받는다.[[12]](#footnote-12) 클라이언트 ~~3~~ 2[[13]](#footnote-13)개가 접속할 때까지 반복한다.**
      * **클라이언트 통신 스레드 – 처음에 ~~어떤 씬 상태인지 검사한다.  
        클라이언트 통신 스레드 – ‘접속 씬’이라면~~ 해당 스레드에 ID를 부여하고 메인 스레드에 ID를 부여한 것을 알린다. 클라이언트에[[14]](#footnote-14) [ID 부여 패킷]을 보내고 [캐릭터 선택 패킷]을 받는다. 이벤트를 이용하여 메인 스레드에 해당 클라이언트가 준비한 것을 알린다.**
      * **메인 스레드 – 모든 클라이언트가 준비했다면 씬 상태를 ‘인 게임 씬’으로 바꾸고 CheckTime() 함수를 이용하여 이전에 이 함수가 호출 되고나서 1/30초가 지나지 않았다면 남은 시간을 기다린다.**
      * **메인 스레드 – ~~이벤트를 다시 리셋하고~~ Update()~~, CollisionCheck()~~ 함수를 진행하고 게임이 종료 되었는지 확인하여 게임이 종료되지 않았다면 CheckTime() 함수부터 반복하고, 게임이 종료 되었다면 모든 클라이언트에 [씬 상태 패킷]이 송신되기를 기다렸다가 서버를 초기화하고 ‘캐릭터 선택 씬’으로 돌아간다. ~~다시 클라이언트의 접속을 받기 시작한다.~~[[15]](#footnote-15)**
      * **~~메인 스레드 – 모든 클라이언트 통신 스레드에 이벤트를 이용하여 데이터를 주고 받으라고 알린다.~~**
      * **클라이언트 통신 스레드 – 메인 스레드로부터 데이터를 주고 받으라는 신호를 기다렸다가 [~~게임 종료 알림~~ 씬 상태[[16]](#footnote-16) 패킷]을 보낸다.**
      * **클라이언트 통신 스레드 – ~~게임이 종료 되었는지~~ 씬 상태를[[17]](#footnote-17) 검사한 후 게임이 종료 되었으면, 스레드를 종료한다. 게임이 종료 되지 않았다면 ~~[캐릭터 크기 패킷],~~ [캐릭터 정보 패킷], [발사체 개수 패킷], [발사체 정보 패킷][[18]](#footnote-18)을 차례로 보낸다.**
      * **클라이언트 통신 스레드 – [플레이어 상태 패킷]을 받고 ~~해당 캐릭터가 발사 시도를 했는지 검사한 후 발사 시도 했다면, [발사 시도 패킷]을 받고 어떤 씬 상태인지 검사하는~~ 플레이어 상태를 갱신한 다음 데이터를 주고 받으라는 신호를 기다리는[[19]](#footnote-19) 부분으로 돌아가 반복한다.**
      * **~~클라이언트 통신 스레드 – 패킷을 받아 서버의 변수를 갱신할 때, 임계영역을 사용한다.~~[[20]](#footnote-20)**

****

**< Server Sequence Diagram >**

**ⓐ ID 부여 이벤트**

**ⓑ 준비 이벤트**

**ⓒ 데이터 주고 받기 이벤트**

**ⓓ [씬 상태 패킷] 보내는 걸 기다리기 위한 이벤트**

* 1. **Client**
     + **클라이언트도 멀티 스레드를 사용해 윈도우 메시지 처리와 소켓 통신을 다른 스레드에서 처리한다.**
     + **메인 스레드 – [서버 통신 스레드]를 생성한다.**
     + **서버 통신 스레드 – 서버에 연결 시도를 하고 연결이 되면 ~~어떤 씬 상태인지 검사한다.~~  
       ~~서버 통신 스레드 – ‘접속 씬’이라면~~[[21]](#footnote-21) [ID 부여 패킷]을 받고 이벤트를 이용해 사용자가 캐릭터 선택을 마칠 때까지 기다렸다가 [캐릭터 선택 패킷]을 보내고 [씬 상태 패킷]을 받아 씬 상태를 갱신한다.[[22]](#footnote-22) ~~‘게임 씬’으로 바꾼 다음 어떤 씬 상태인지 검사하는 부분으로 돌아간다.~~**
     + **서버 통신 스레드 – ~~‘게임 씬’이라면 [게임 종료 알림 패킷]을 받고~~[[23]](#footnote-23) 게임이 종료 되었는지 검사하여 종료 했다면 ~~스레드를 종료한다.~~ 게임 종료 화면을 출력하고 게임을 초기화하여 ‘캐릭터 선택 씬’으로 돌아간다.[[24]](#footnote-24)**
     + **서버 통신 스레드 – 게임이 종료 되지 않았다면, ~~[캐릭터 사이즈 패킷],~~ [캐릭터 정보 패킷], [발사체 개수 패킷], [발사체 정보 패킷][[25]](#footnote-25)을 순차적으로 받고 [플레이어 상태 패킷]을 보내고 ~~플레이어가 발사 시도를 했다면 [발사 시도 패킷]을 보내고~~ ~~어떤 씬 상태인지 검사하~~ [씬 상태 패킷]을 받[[26]](#footnote-26)는 부분으로 돌아가 반복한다.**
     + **메인 스레드 - 키보드와 마우스 입력이 있을 때마다 자신의 상태를 갱신한다.**
     + **메인 스레드 – WM\_PAINT 메시지 호출이 있을 때마다 화면을 새로 출력한다.**

**스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명**

**< Client Sequence Diagram >**

**ⓐ 캐릭터 선택 이벤트**

**ⓑ 게임 종료 화면을 넘기는 마우스 입력 확인 이벤트**

1. **Low-level Design**
   1. **Server**

* **서버 초기화 함수**

**void Initialize\_Server();**

* **서버 리셋 함수**

**void ResetServer();[[27]](#footnote-27)**

* **클라이언트 통신 스레드 함수**

**DWORD WINAPI ComClient(LPVOID arg);**

* **클라이언트와 1초당 30번의 데이터를 주고 받기 위해 서버의 시간을 체크해 주는 함수**

**void CheckTime(DWORD Time);**

* **메인 스레드 안에서 ~~캐릭터와 발사체의 위치 정보를 갱신해주는~~ 1/30 초 동안 서버에서 해야 할 작업들이 모두 들어있는 함수**

**void Update(DWORD TickTime);**

* **~~충돌 체크 검사 및 처리 함수~~**

**~~void CollisionCheck();~~**

* **캐릭터 <-> 장애물 충돌 검사 함수**

**bool CC\_Char\_Obst(S\_Character\*, S\_Obstacle\*);**

* **캐릭터 <-> 캐릭터 충돌 검사 함수**

**bool CC\_Char\_Char(S\_Character\*, S\_ Character\*);**

* **캐릭터 <-> 발사체 충돌 검사 함수**

**bool CC\_Char\_Bullet(S\_Character\*, S\_ Bullet\*); [[28]](#footnote-28)**

* 1. **Client**
     + **클라이언트 초기화 함수  
       ~~void Initialize\_Client();~~  
       SceneManager::SceneManager();[[29]](#footnote-29)**
     + **클라이언트 리셋 함수**

**void SceneManager::ResetClient();[[30]](#footnote-30)**

* + - **서버 통신 스레드 함수**

**DWORD WINAPI ComServer(LPVOID arg);**

* + - **키 입력 처리 함수**

**void ProcessKeyDown(WPARAM);**

**void ProcessKeyUp(WPARAM);[[31]](#footnote-31)**

* 1. **Enum, 구조체, 패킷**
* **enum ObjectsType { BOMBER, GUNNER, MAGICIAN, DIE, BOOM, BULLET, FIRE, NONE[[32]](#footnote-32) };**
* **enum DirectionState { DOWN\_LEFT, LEFT, UP\_LEFT, DOWN, UP, DOWN\_RIGHT, RIGHT, UP\_RIGHT };[[33]](#footnote-33)**
* **enum SceneState { SELECT\_CHARACTER, IN\_GAME, PLAYER1WIN, PLAYER2WIN, PLAYER3WIN };[[34]](#footnote-34)**
* **struct vector2**

**{**

**short X;**

**short Y;**

**}**

* **[ 서버 -> 클라이언트 ] ID 부여 패킷**

**struct PACKET\_ClientID**

**{**

**BYTE ClientID**

**}**

* **[ 서버 -> 클라이언트 ] ~~게임 종료 알림~~ 씬 상태 패킷**

**struct PACTET\_~~GameOver~~ SceneState**

**{**

**BYTE ~~IsGameOver~~ SceneState;**

**}[[35]](#footnote-35)**

* **~~[ 서버 -> 클라이언트 ] 캐릭터 사이즈 패킷~~**

**~~struct PACKET\_CharacterPacketSize~~**

**~~{~~**

**~~unsigned short Size;~~**

**~~}~~[[36]](#footnote-36)**

* **[ 서버 -> 클라이언트 ] 캐릭터 정보 패킷**

**struct PACKET\_Character**

**{**

**vector2 Position;**

**short Life;**

**BYTE Type;**

**BYTE AnimationState;**

**BYTE AnimationCount;**

**~~STRUCT\_Bullet\* Bullets;~~[[37]](#footnote-37) }**

* **[ 서버 -> 클라이언트 ] 발사체 개수 패킷**

**struct PACKET\_BulletNum**

**{**

**BYTE BulletNum;**

**}**

* **~~캐릭터 정보 패킷 안에 들어가는 발사체 구조체~~  
  [ 서버 -> 클라이언트 ] 발사체 정보 패킷**

**struct ~~STRUCT~~ PACKET\_Bullet[[38]](#footnote-38)**

**{**

**vector2 Position;**

**BYTE Type;**

**BYTE AnimationCount;**

**}**

* **[ 클라이언트 -> 서버 ] 캐릭터 선택 패킷**

**struct PACKET\_ChoiceCharacter**

**{**

**BYTE CharacterType;**

**}**

* **[ 클라이언트 -> 서버 ] 플레이어 상태 패킷**

**struct PACKET\_PlayerState**

**{**

**BYTE DirectionState;**

**BYTE IsMoving;**

**~~BYTE AttackAttempt;~~**

**~~vector2 AttackDirection;~~**

**float AttackDirectionX;**

**float AttackDirectionY;**

**}[[39]](#footnote-39)**

* **~~[ 클라이언트 -> 서버 ] 발사 시도 패킷~~**

**~~struct PACKET\_AttackAttempt~~**

**~~{~~**

**~~vector2 Direction;~~**

**~~}~~[[40]](#footnote-40)**

* **발사 시도 구조체**

**struct AtteckDirection**

**{**

**float X;**

**float Y;**

**}[[41]](#footnote-41)**

1. **팀원 별 역할분담**

|  |  |
| --- | --- |
| **팀원** | **역할** |
| **배강산 (PM)** | **프로젝트 전체적인 일정을 관리하고 클라이언트와 서버의 데이터 송수신 부분, 캐릭터와 발사체 위치 업데이트, 서버의 충돌검사 및 처리, 발사체의 생성 구현.** |

1. **개발 환경**

* **Visual Studio 2017**
* **C / C++**
* **Win API**
* **Tiled**
* **Git[[42]](#footnote-42)**
* **Photoshop[[43]](#footnote-43)**

1. **개발일정**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 월 | 화 | 수 | 목 | 금 | 토 | 일 |
|  |  | 11/22  서버 프로젝트 생성, 캐릭터 클래스 추가 | 23  서버에 Initialize\_  Server() 함수 추가, 발사체 클래스 추가 | 24  서버에ComClient() 스레드 함수 추가, 준비 상태를 확인하기 위한 이벤트 추가 | 25, 26  미 구현 부분 추가 구현  프로젝트  테스트 | |
| 27  클라이언트에 캐릭터 선택하여 준비하는 부분 구현  ProcessKeyDown(), ProcessKeyUp() | 28  서버에 모든클라이언트의 준비 상태를 확인해 게임 씬으로 변경하는 부분 구현  WaitForMultipleObjects(hEventReady)  SetEvent(hEventReady) | 29  서버에서 클라로 [~~게임 종료 알림~~ 씬 상태 패킷] 보내는 부분 구현 | 30  서버와 클라에 ~~게임 종료~~ 어떤 씬인지 확인하는 부분 구현 | 12/1  서버에 Update() 함수 구현  클라이언트에 플레이어 상태 정의하는 부분 디버깅 | 2, 3  미 구현 부분 추가 구현  프로젝트  테스트 | |
| 4  ~~서버에 CollisionCheck() 함수 구현~~  충돌 검사 구현  CC\_Char\_Obst(), CC\_Char\_Char(), CC\_Char\_Bullet() | 5  ~~CollisionCheck()~~ Update() 함수 내에 충돌 처리 내용 구현 | 6  서버에 ~~발사 시도가 정당한지 확인하여~~ 발사체 생성하는 부분 구현 SetBulletState () | 7  클라에서 서버로 [플레이어 상태 패킷]~~, [발사 시도 패킷]~~ 보내는 부분 구현 | 8  서버에서 클라이언트로 ~~[캐릭터 크기 패킷],~~ [캐릭터 정보 패킷], [발사체 개수 패킷], [발사체 정보 패킷] 보내는 부분 구현 | 9, 10  미 구현 부분 추가 구현  프로젝트  테스트 | |
| 11  클라에 배경 이미지 적용,  각 캐릭터 간 밸런스 조정 | 12  최적화 작업 | 13  디버깅 및  프로젝트  테스트 | 14  디버깅 및  프로젝트  테스트 | 15  디버깅 및  프로젝트  테스트 | 16, 17  디버깅 및  프로젝트  테스트 | |
| 18  최종 검수 |  |  |  |  |  | |

1. **Git Commit log**

**$ git log --reverse**

**commit 6511cd7287efdf30f67cc662d8746d1f1a021627**

**Author: warimo47 <warimo47@naver.com>**

**Date: Tue Nov 21 15:59:52 2017 +0900**

**Initial commit**

**commit a6172f9eea7d512ba332275a4bfa4e5918b9d4f4**

**Author: kangsan <warimo47@naver.com>**

**Date: Tue Nov 21 16:02:59 2017 +0900**

**First commit**

**commit d1bee8a6c4dc7eba8059fddb82c90fdd3e8a24e6**

**Author: kangsan <warimo47@naver.com>**

**Date: Tue Nov 21 19:08:17 2017 +0900**

**추진 계획서 작성**

**commit 0d94f75cd3f6b8933e85c13aab19f85892059f48**

**Author: kangsan <warimo47@naver.com>**

**Date: Wed Nov 22 15:23:24 2017 +0900**

**Add NPP Server Project**

**Change report**

**1. Sequence diagram**

**2. delete AttackAttempt packet**

**3. Service 2 Clients**

**commit 0e8aaf8d0c8fcdd6f5f2ddfb8b325d62da3967c5**

**Author: kangsan <warimo47@naver.com>**

**Date: Wed Nov 22 15:26:38 2017 +0900**

**delete .vs, Debug folder**

**commit c855021dccaeba98550b99f9552d24b33d74a2e1**

**Author: kangsan <warimo47@naver.com>**

**Date: Wed Nov 22 16:33:42 2017 +0900**

**Server:**

**Client:**

**Add Packet struct**

**commit f176038cfb79fcd76477ff90eed2de31df90fa13**

**Author: kangsan <warimo47@naver.com>**

**Date: Wed Nov 22 18:11:55 2017 +0900**

**Change NGP Client Project**

**commit f74de134dfe91b5bef7db186708713c85c65362e**

**Author: kangsan <warimo47@naver.com>**

**Date: Wed Nov 22 19:08:03 2017 +0900**

**서버와 클라이언트 연결을 확인함**

**commit 993291598dbb9d983ea5e445bfbda8d1c24225f6**

**Author: kangsan <warimo47@naver.com>**

**Date: Wed Nov 22 19:56:00 2017 +0900**

**Change NGP Server**

**Add S\_Character struct**

**S\_Bullet struct**

**commit fe4fcf076e644c7febe355acfdad8b1568870d03**

**Author: kangsan <warimo47@naver.com>**

**Date: Wed Nov 22 19:56:56 2017 +0900**

**.**

**commit 46e217347d249f479e58ae7e6a08e96f7304cf59**

**Author: kangsan <warimo47@naver.com>**

**Date: Wed Nov 22 20:26:12 2017 +0900**

**클라이언트에 씬 나누고 선택 할 캐릭터 그리기**

**commit ffc81e9b665f517d08b07a56e65452cf6798d0ee**

**Author: kangsan <warimo47@naver.com>**

**Date: Wed Nov 22 23:21:38 2017 +0900**

**클라이언트 : 캐릭터 선택 구현**

**commit ed675f12eacee6dbaf2fd7ae527770aed489e238**

**Author: kangsan <warimo47@naver.com>**

**Date: Thu Nov 23 11:19:51 2017 +0900**

**ID 부여 패킷 송수신**

**캐릭터 선택 패킷 송수신**

**인 게임으로 씬 전환**

**commit 5792ccb9eee1c2e63bd1e8ece8bc1b9a7197f248**

**Author: kangsan <warimo47@naver.com>**

**Date: Thu Nov 23 11:50:35 2017 +0900**

**.**

**commit e804da3f0c1366165c558e5a4d5a9dd28000498b**

**Author: kangsan <warimo47@naver.com>**

**Date: Thu Nov 23 18:54:57 2017 +0900**

**맵 변경**

**commit 8a65f1f2f409fcede8b3a6f8dddfbb98183c8c80**

**Author: kangsan <warimo47@naver.com>**

**Date: Fri Nov 24 15:47:10 2017 +0900**

**서버에서 게임을 시작하면 모든 클라이언트가 같이 시작하게 함**

**commit 43d790c3b2b338d4f973e541b2be3683c6fcf59b**

**Author: kangsan <warimo47@naver.com>**

**Date: Mon Nov 27 11:23:37 2017 +0900**

**Debug folder Upload**

**commit 747981fc5b876a349c52237101923550b8a394eb**

**Author: kangsan <warimo47@naver.com>**

**Date: Mon Nov 27 11:30:22 2017 +0900**

**.**

**commit 04b3efd2e21fac8bfa0980d8ec24806a5159969e**

**Author: kangsan <warimo47@naver.com>**

**Date: Mon Nov 27 12:47:07 2017 +0900**

**delete .vs, Debug**

**commit fe848213c63b192ee2edb1a6ae283fd03d63cfce**

**Author: kangsan <warimo47@naver.com>**

**Date: Mon Nov 27 16:58:43 2017 +0900**

**추진 계획서**

**시퀀스 다이어그램 수정**

**commit e76e4e3e174666c3f59ea5778f5048158f3e41d0**

**Author: kangsan <warimo47@naver.com>**

**Date: Mon Nov 27 21:42:45 2017 +0900**

**서버에서 클라이언트로 캐릭터 정보 보내는 부분 구현**

**commit bbd02c7db2451563ed90928c541186e3b5b2472c**

**Author: kangsan <warimo47@naver.com>**

**Date: Tue Nov 28 14:41:58 2017 +0900**

**접속시 IP 주소 입력 받도록 수정**

**commit 7eee99c0e4a87147b82b117961dd16ff36966442**

**Author: kangsan <warimo47@naver.com>**

**Date: Tue Nov 28 22:48:22 2017 +0900**

**키 입력 처리 추가**

**commit f5176b209271eff7684be2b858f9787496a8f8aa**

**Author: Bae Kang San <warimo47@kpu.ac.kr>**

**Date: Wed Nov 29 13:03:21 2017 +0900**

**시간 개념 도입**

**commit 3c7e702628b18393544bdb813b016952df2c19b6**

**Author: kangsan <warimo47@naver.com>**

**Date: Thu Nov 30 19:59:56 2017 +0900**

**..**

**commit 329daba862740b221ceb93c044dcd711866c32a3**

**Author: Bae Kang San <warimo47@kpu.ac.kr>**

**Date: Thu Nov 30 21:36:20 2017 +0900**

**캐릭터 움직임 구현**

**commit dc54fc03a53a957a966b18eef35c66f66f7271ab**

**Author: kangsan <warimo47@naver.com>**

**Date: Fri Dec 1 17:51:46 2017 +0900**

**발사체 이미지 추가**

**commit 936776893034daa050ab122f5136c1425a882829**

**Author: kangsan <warimo47@naver.com>**

**Date: Fri Dec 1 18:52:44 2017 +0900**

**발사체 발사 성공**

**commit 35a06fdf68cc6c3272bd3eeb166686c962f4bc62**

**Author: kangsan <warimo47@naver.com>**

**Date: Fri Dec 1 21:34:59 2017 +0900**

**발사 방향 값이 0,0 이 들어감**

**commit e9dcb658643e972219fde278343719bc3eb1815f**

**Author: Bae Kang San <warimo47@kpu.ac.kr>**

**Date: Sat Dec 2 19:59:40 2017 +0900**

**..**

**commit 6bcd13d21eb1ac98f89715f8f8bbabc7e17ae408**

**Author: kangsan <warimo47@naver.com>**

**Date: Mon Dec 4 10:12:06 2017 +0900**

**발사체 궤도 오차 수정**

**commit 50274501ab4f1f42dc077d07d12b603a67a1991d**

**Author: kangsan <warimo47@naver.com>**

**Date: Mon Dec 4 10:17:45 2017 +0900**

**추진 계획서 수정**

**commit f7d885e0c9cc36d2e57c553f064e680b9f2bc212**

**Author: Bae Kang San <warimo47@kpu.ac.kr>**

**Date: Mon Dec 4 12:03:45 2017 +0900**

**캐릭터 <-> 캐릭터 충돌 검사 구현**

**commit 2e4d7d810baede7d4de492b45a5d31add0eeb1b6**

**Author: kangsan <warimo47@naver.com>**

**Date: Mon Dec 4 16:39:10 2017 +0900**

**충돌 체크 및 충돌 처리 구현**

**commit cb9e1bb60701675e01f298da7fa30c779c430479**

**Author: Bae Kang San <warimo47@kpu.ac.kr>**

**Date: Wed Dec 6 12:46:51 2017 +0900**

**플레이어 어느 한쪽이 죽으면 게임 종료되게 구현**

**commit bb1bc11e1b616f28b26d64814ea777b29850052c**

**Author: kangsan <warimo47@naver.com>**

**Date: Wed Dec 6 20:44:46 2017 +0900**

**캐릭터 선택 장면 배경 이미지 변경**

**commit cf581671f6059de77bab3be5a437ac1a42a2b581**

**Author: kangsan <warimo47@naver.com>**

**Date: Thu Dec 7 17:25:55 2017 +0900**

**게임이 종료되면 캐릭터 선택으로 돌아감**

**commit d320cfa325ad8d0a36db0c0fa93741d69c7f7539**

**Author: Bae Kang San <warimo47@kpu.ac.kr>**

**Date: Sun Dec 10 22:53:46 2017 +0900**

**장애물 추가**

**commit 284ae7103603f7b24982d179dfbc9dc2ad30e5ac**

**Author: Bae Kang San <warimo47@kpu.ac.kr>**

**Date: Mon Dec 11 11:22:13 2017 +0900**

**재시작 시 캐릭터, 발사체 리셋 구현**

**commit ab870d8e5cf9c0fce87d29615c91bfaf6a67b05d**

**Author: Bae Kang San <warimo47@kpu.ac.kr>**

**Date: Mon Dec 11 12:20:06 2017 +0900**

**승리 패배 화면 구현**

**commit 5bc6b6e5646d0fa58cc03f44722e2befdf30148c**

**Author: kangsan <warimo47@naver.com>**

**Date: Mon Dec 11 20:57:15 2017 +0900**

**캐릭터 발사체 애니메이션 구현**

**commit 19be4f9245e83b7552316a605a4d9af716a33f3f**

**Author: kangsan <warimo47@naver.com>**

**Date: Tue Dec 12 12:56:06 2017 +0900**

**쿨타임 변경 적용**

**commit 0a14dec704eb569d06fdcc65338b38588ef22733**

**Author: kangsan <warimo47@naver.com>**

**Date: Tue Dec 12 19:57:52 2017 +0900**

**추진 계획서 수정 중**

**commit 597ed3d951ab8ca138f0c5cbf1ef616044ccd586**

**Author: kangsan <warimo47@naver.com>**

**Date: Tue Dec 12 22:35:38 2017 +0900**

**추진 계획서 Low level, 일정 수정 필요**

**commit 179dbb21a37c867e53fbc9a299babc7726805de4 (HEAD -> master, origin/master, origin/HEAD)**

**Author: kangsan <warimo47@naver.com>**

**Date: Wed Dec 13 11:26:15 2017 +0900**

**추진 계획서 수정 중**

**(END)**

<https://github.com/warimo47/NGPTermProject>

1. 클라이언트 개수 3개 -> 2개. [↑](#footnote-ref-1)
2. 클라이언트 개수 3개 -> 2개. [↑](#footnote-ref-2)
3. 이름 변경. [↑](#footnote-ref-3)
4. [발사 시도 패킷]의 내용을 [플레이어 상태 패킷] 안에 넣어서 합침. [↑](#footnote-ref-4)
5. [발사 시도 패킷]의 내용을 [플레이어 상태 패킷] 안에 넣어서 합침. [↑](#footnote-ref-5)
6. 캐릭터가 장애물 뒤에서 발사체를 쏘면 바로 없어지는 예상하지 못한 상황이 발생하여 기획 변경. [↑](#footnote-ref-6)
7. [발사 시도 패킷]의 내용을 [플레이어 상태 패킷] 안에 넣어서 합침. [↑](#footnote-ref-7)
8. 캐릭터와 발사체를 각기 다른 클래스로 구분함. [↑](#footnote-ref-8)
9. 캐릭터 클래스와 발사체 클래스를 구분하면서 화면에 그리는 것도 구분해서 그리게 수정. [↑](#footnote-ref-9)
10. 클라이언트 개수 3개 -> 2개. [↑](#footnote-ref-10)
11. 클라이언트 개수 3개 -> 2개. [↑](#footnote-ref-11)
12. 스레드가 새로 생성되고 ID 부여 받기까지 기다리지 않으면 똑같은 ID를 부여 받는 클라이언트가 생길 수 있는 문제 해결을 위해 추가 됨. [↑](#footnote-ref-12)
13. 클라이언트 개수 3개 -> 2개. [↑](#footnote-ref-13)
14. 메인 스레드가 ID 부여하는 것을 기다리고 있기 때문에 이벤트를 Set해주어 ID 부여된 것을 알림. [↑](#footnote-ref-14)
15. 게임이 종료되면 클라이언트의 연결을 다시 받던 기존의 기획에서 게임이 종료되면 ‘캐릭터 선택 씬’으로 돌아가게 변경. (비효율적인 기획 내용의 변경) [↑](#footnote-ref-15)
16. 패킷의 이름 변경. [↑](#footnote-ref-16)
17. 문장의 표현 변경. [↑](#footnote-ref-17)
18. 캐릭터와 발사체를 클래스를 구분해서 관리하여 패킷도 구분함. [↑](#footnote-ref-18)
19. 기존의 추진계획서가 잘못 작성되어 있었던 부분 수정. [↑](#footnote-ref-19)
20. 공유 변수를 두 스레드에서 동시에 수정하는 부분이 없게 구성하여 임계영역을 사용하지 않음. [↑](#footnote-ref-20)
21. 불필요한 검사를 하는 부분 삭제. [↑](#footnote-ref-21)
22. 모든 클라이언트가 인 게임으로 넘어가는 시간을 동기화하기 위해 [씬 상태 패킷]을 받도록 변경. [↑](#footnote-ref-22)
23. 불필요한 검사를 하는 부분 삭제. [↑](#footnote-ref-23)
24. 게임이 종료되면 클라이언트의 연결을 다시 받던 기존의 기획에서 게임이 종료되면 ‘캐릭터 선택 씬’으로 돌아가게 변경. (비효율적인 기획 내용의 변경) [↑](#footnote-ref-24)
25. 캐릭터와 발사체를 각기 다른 클래스로 구분하면서 패킷도 구분하여 보내게 변경. [↑](#footnote-ref-25)
26. [발사 시도 패킷]의 내용을 [플레이어 상태 패킷] 안에 넣어서 합침. [↑](#footnote-ref-26)
27. 게임이 종료되면 새로운 게임의 캐릭터 선택을 하게 기획을 변경하며 추가된 함수. [↑](#footnote-ref-27)
28. 충돌 검사하는 함수가 3개로 나누어짐. [↑](#footnote-ref-28)
29. 클라이언트에서 수행하는 작업을 SceneManager 클래스 내에서 하도록 구현. [↑](#footnote-ref-29)
30. 게임이 종료되면 캐릭터 선택으로 돌아가는 기획 변경 사항 때문에 추가된 함수. [↑](#footnote-ref-30)
31. 키 입력을 처리하는 코드 내용이 길어져서 따로 함수로 구분함. [↑](#footnote-ref-31)
32. 발사체 배열에서 발사체가 없는 것을 표현하기 위한 enum 추가. [↑](#footnote-ref-32)
33. 플레이어의 방향 상태를 결정하기 위한 enum 추가. [↑](#footnote-ref-33)
34. 씬 상태를 표현하기 위한 enum 추가. [↑](#footnote-ref-34)
35. 이름 변경. [↑](#footnote-ref-35)
36. 캐릭터 정보 패킷의 크기가 가변 길이에서 고정 길이로 변경되어 불필요하기 때문에 삭제. [↑](#footnote-ref-36)
37. [발사체 정보 패킷]을 분리함. [↑](#footnote-ref-37)
38. 캐릭터와 발사체를 분리하면서 [발사체 정보 패킷]을 [캐릭터 정보 패킷]에서 분리해 냄. [↑](#footnote-ref-38)
39. vector2 구조체는 short 형 변수를 가지고 있어 실수 표현이 불가능하므로 float 형태로 변경. [↑](#footnote-ref-39)
40. 불필요한 패킷 삭제. [↑](#footnote-ref-40)
41. 각각의 플레이어의 발사 시도 상태 값을 서버에서 저장해서 관리하기 위한 구조체 추가. [↑](#footnote-ref-41)
42. 프로젝트 버전 관리를 위해 추가. [↑](#footnote-ref-42)
43. 리소스 편집에 사용함. [↑](#footnote-ref-43)