**2023 NGP 2팀**

**네트워크 게임 프로그래밍**

**프로젝트 추진계획서**

2019182001 강승호

2021182001 강은혁

2021182022 엄미영

**목차**

1. 게임소개
   1. 게임 소개
   2. 게임 요소
   3. 게임 진행과 전개
2. 서버-클라 차트
   1. High-level
   2. Low-level
3. 개인 일정 및 진척도
   1. 개인별 일정
   2. 진척도 표시

**1. 게임 소개**

**1.1. 게임 소개**

**1) 게임 장르**

▷ 2D 슈팅 캐주얼 게임

**2) 게임 토대 및 내용**

▷ 윈도우 프로그래밍 시간에 만들었던 게임으로 기존 총알로 몬스  
 터를 처리하는 플레이 방식을 바꿔 플레이어들끼리 싸워 상대를   
 탈락시키는 게임이다.

**1.2. 게임 요소**

**1) 플레이어 수** : 2 ~ 5명

**2) 조작**

▷ 방향키(조작)

▷ D (공격)

**3) 게임 화면**

▷ 전체 맵 : 1200 x 1200

▷ 카메라 : 800 x 800, 플레이어 중심

▷ 프레임 : 60

**4) 플레이어**

▷ 탱크형, 플레이어마다 색이 다름.

▷ 체력 : 100

▷ 이동속도 : 3 (프레임)

▷ 크기 : 30 x 30

**5) 총알**

▷ 공격력 : 6

▷ 속도 : 10 (프레임)

▷ 플레이어와 충돌하거나 화면 밖으로 벗어나면 없어짐

**6) 아이템**

▷ 획득 즉시 플레이어에게 효과 적용됨.

▷ 아이템은 5초마다 랜덤위치에 랜덤 종류로 나타남

▷ 얼음 : 플레이어 과열 게이지 0으로 만들어줌

▷ 주황보석 : 이동속도 +10%

▷ 빨간보석 : 플레이어 공격력 + 2

▷ 골드 : 점수 5점

**7) UI**

▷ 좌측 상단 : 모든 플레이어의 HP

▷ 좌측 하단 : 본인의 HP와 과열 게이지, 점수

▷ 중앙 상단 : 남은 시간

**1.3. 게임 진행과 전개**

**1) 게임 진행 방식과 규칙**

▷ 플레이어는 게임 시작과 동시에 다른 플레이어를 기다린다.

▷ 플레이어가 2명 이상 있으면 게임을 시작할 수 있다.

▷ 게임이 시작되면 플레이어는 아이템을 먹으며 다른 플레이어를  
 처치 한다.

▷ 플레이어는 한 발을 쏠 때마다 과열 수치가 올라가는데 10발을   
 쓰게 되면 과열 수치가 최대가 되어 수치가 다시 떨어질 때까지   
 총알을 발포할 수 없다. ( 1초마다 2씩 떨어짐 )

▷ 게임 내에는 다양한 아이템이 존재하며 플레이어는 이를 획득해  
 어드벤티지를 얻을 수 있다.

▷ 최후 1인이 남거나 제한 시간 2분 내에 점수를 많이 얻은 사람  
 이 승리한다. 게임 승자가 정해지면 게임은 종료된다.

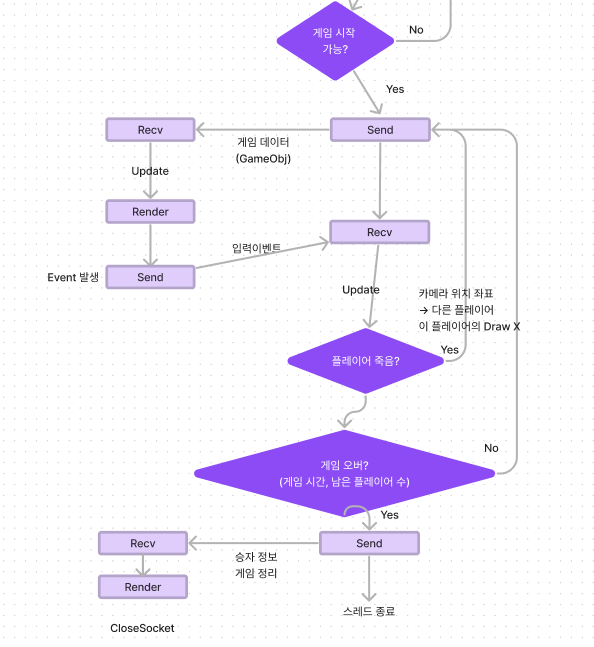
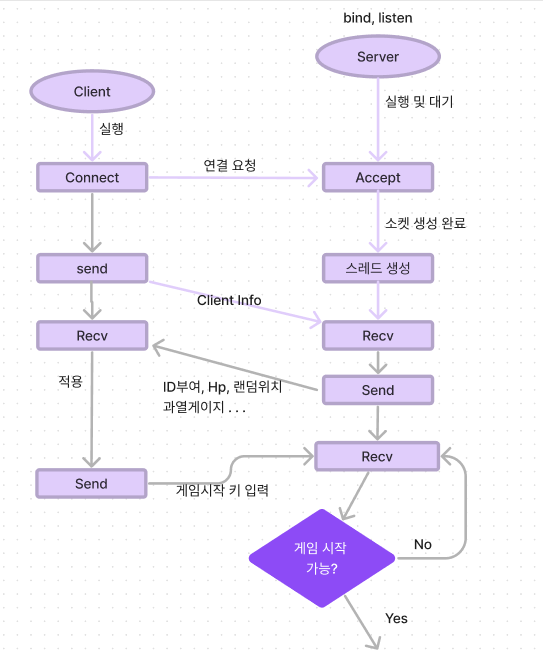
▷ 게임 점수는 맵 내의 코인과 플레이어가 처치한 상대 플레이어  
 를 합산 수치이다. (코인 5점 + 처치 30점)

▷ 사망한 플레이어는 관전모드로 들어간다.

▷ 게임 시작할 때 랜덤 위치로 장애물과 아이템이 생성된다.

**2. 서버-클라 차트**

**2.1. High-level**



**2.2. Low-level**

**(추후 클래스 및 패킷, 함수 정리 필요)**  
**<공동>**  
enum MSG  
{  
MSG\_ WAITING - 플레이어 로비상태(게임 시작 전) 메세지  
MSG\_GAME - 플레이어 인게임 상태 메세지  
MSG\_COLLIDE - 충돌 메세지  
MSG\_LEAVE - 플레이어 이탈 메세지  
MSG\_CLEAR - 게임 클리어 메세지  
}  
  
void sendMSG() - 메세지 정보 송신  
void recvMSG() - 메세지 정보 수신  
  
  
**< 서버 >**  
  
#define Death 0  
#define Alive 1  
  
void IsGameStartAble() // 게임 시작 가능 여부 함수

void CollisionCheck(); // 충돌처리 함수

class Object // 오브젝트 클래스

class Item // 아이템클래스

class Player // 플레이어 클래스  
void IsPlayerDead(); // 플레이어 사망 여부 함수  
void GameOver(); // 게임오버 함수 (시간과 플레이어 수 처리)  
void ElapsedTime(); // 시간 계산 함수  
void CreateItems(); // 아이템 생성 함수  
void CreateObstacles(); // 장애물 생성 함수  
  
  
- 송신  
void SC\_sendPlayerData(); ->클라에게 플레이어 기본 게임 데이터 보냄  
void SC\_sendObjectData(); -> 클라에게 게임 오브젝트 데이터 전달  
void SC\_sendPlayerUpdate(); -> 클라에게 모든 클라이언트의 ID, 위치, 과열 게이지 등 플레이어 정보 알림  
void SC\_sendServerTime(); -> 게임 진행시간을 보냄  
(교수님) void SC\_sendPlayerDeath() -> 죽은 클라이언트 상태를 보냄 ( 다른 플레이어의 카메라 위치를 이 클라이언트 플레이어의 좌표로 삼되 플레이어를 그리지 않는다?)   
void SC\_sendGameClear() -> 클라에게 게임 클리어 여부를 보냄 (우승 정보->점수와 등수 계산, 정리해야 할 게임 정보)  
  
- 수신  
void CS\_recvPlayerInIt(); -> 클라로부터 정보 받음  
void CS\_recvGameStart(); -> 클라로부터 게임 시작 키 입력 받아 게임이 시작 가능한지 체크  
void CS\_recvPlayerInfo(); -> 클라로부터 플레이어 키 입력 값 받음  
  
**< 클라이언트 >**  
void KeyEvent(); //클라이언트 키 입력함수  
void UpdateGameState();// 게임 업데이트 함수  
void GameRender();// 게임 렌더 함수  
void ShowWinner();// 우승자 정보 띄우기 함수  
void EndGame(); // 게임 정리 함수  
  
- 송신  
void CS\_sendPlayerInIt(); -> 서버에게 클라 정보 보냄  
void CS\_sendGameStart(); -> 게임시작이벤트 전달  
void CS\_sendPlayerInfo(); -> 서버에게 클라의 ID와 키 입력 값 알림   
  
- 수신  
void SC\_recvPlayerData(); -> 서버로부터 게임 데이터 받음  
void SC\_recvObjectData(); -> 서버로부터 오브젝트 받음  
void SC\_recvPlayerUpdate(); -> 서버로부터 플레이어의 정보를 받아 update로 전달  
void SC\_recvServerTime(); --> 서버로부터 게임 진행시간을 받음  
void SC\_recvGameClear() -> 서버로부터 게임이 종료됨을 받음

**3. 개인 일정 및 진척도**

**3.1. 역할 분담**

**- 강승호 : 클라이언트**

* 1. 기존 게임에 UI - 시간, 전체 플레이어 점수 구현
  2. 기본 네트워크 환경 구축 (기본적인 클라 구현)
  3. 로비 클라이언트 제작
  4. 클라이언트 통신 구조체 작성
  5. 클라이언트에서 정보 송신하는 부분 제작(키 송신 및 이벤트)
  6. 서버 데이터를 받아 렌더링
  7. 서버에서 플레이어 간의 충돌처리
  8. 게임 마무리 함수
* **- 강은혁 : 서버**
  1. 기존 게임에 아이템 부분 구현
  2. 기본 네트워크 환경 구축 (기본적인 서버 구현)
  3. 로비의 서버 부분 제작
  4. 클라이언트의 송수신 함수
  5. 아이템 - 플레이어 충돌처리
* **- 엄미영 : 서버**
  1. 기존 게임 코드 클래스/파일 분리 처리
  2. 기본 네트워크 환경 구축 (기본적인 서버 구현)

3. 메세지 식별 함수 구현

4. 플레이어 아이디 부여 및 사망 처리 구현

5. 플레이어 - 벽, 총알 충돌처리

6. 서버의 송수신 함수

**3.2. 개인별 일정**

- 강승호

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **일** | | **월** | | **화** | **수** | | **목** | | **금** | | | **토** |
|  | | 10/30 | | 10/31 | 11/1 | | 11/2 | | 11/3 | | | 11/4 |
|  | | 추진서  제출 | |  | 시간, 플레이어 점수 UI 구현 | |  | | 로비부분 제작 | | |  |
| 11/5 | | 11/6 | | 11/7 | 11/8 | | 11/9 | | 11/10 | | | 11/11 |
| 키 이벤트 처리함수  통신 구조체 작성 | | 중간고사 | | 중간고사 | 중간고사 | | 플레이어 정보 전송, 시작이벤트 전달 함수 | | 클라이언트에게 키입력값, ID 알림함수 | | |  |
| 11/12 | | 11/13 | | 11/14 | 11/15 | | 11/16 | | 11/17 | | | 11/18 |
| **서버로부터 게임데이터 수신 함수** | |  | | 중간고사 |  | |  | | 게임 진행시간 수신 함수 | | |  |
| 11/19 | | 11/20 | | 11/21 | 11/22 | | 11/23 | | 11/24 | | | 11/25 |
|  | | 서버의 플레이어간 충돌처리 | |  |  | | 서버로부터 오브젝트 수신 함수 | | 모든플레이어 정보 받는 함수 | | |  |
| **일** | **월** | | **화** | | | **수** | | **목** | | **금** | **토** | |
| 11/26 | 11/27 | | 11/28 | | | 11/29 | | 11/30 | | 12/1 | 12/2 | |
|  | 승자 출력 구현 | | 게임종료수신함수, 게임 마무리 함수 | | |  | |  | | 테스트 및 마무리 | 테스트 및 마무리 | |
| 12/3 |  | |  | | |  | |  | |  |  | |
| 테스트 및 마무리 |  | |  | | |  | |  | |  |  | |

- 강은혁

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **일** | **월** | **화** | **수** | **목** | **금** | **토** |
|  | 10/30 | 10/31 | 11/1 | 11/2 | 11/3 | 11/4 |
|  | 추진서  제출 |  |  |  | 아이템 클래스 구현 |  |
| 11/5 | 11/6 | 11/7 | 11/8 | 11/9 | 11/10 | 11/11 |
|  | 졸작 회의 | 충돌 처리 구현 |  |  | 서버 로비 처리 함수 구현 |  |
| 11/12 | 11/13 | 11/14 | 11/15 | 11/16 | 11/17 | 11/18 |
|  |  | 시험 | **클라이언트 플레이어 데이터 송신 함수 구현** |  | **서버 플레이어 데이터 수신 함수 구현** |  |
| 11/19 | 11/20 | 11/21 | 11/22 | 11/23 | 11/24 | 11/25 |
|  | 졸작 회의 | 서버의 오브젝트 데이터 송신 함수 구현 |  |  | **클라이언트의 오브젝트 데이터 수신 함수 구현** |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **일** | **월** | **화** | **수** | **목** | **금** | **토** |
| 11/26 | 11/27 | 11/28 | 11/29 | 11/30 | 12/1 | 12/2 |
| 서버 업데이트 함수 구현 |  | 클라이언트 업데이트 함수 구현 |  |  | 데모 및 버그 수정 | 데모 및 버그 수정 |
| 12/3 |  |  |  |  |  |  |
| 데모 및 버그 수정 |  |  |  |  |  |  |

- 엄미영

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **일** | **월** | **화** | **수** | **목** | **금** | **토** |
|  | 10/30 | 10/31 | 11/1 | 11/2 | 11/3 | 11/4 |
|  | 추진서  제출 | 졸작회의 클래스/  파일분리 | 네트워크  분리 작업 (기본 토대) | 네트워크 기본 환경 구축 | 졸작회의  네트워크 기본 환경 구축 | 네트워크 기본 환경 구축 |
| 11/5 | 11/6 | 11/7 | 11/8 | 11/9 | 11/10 | 11/11 |
|  | 메세지 식별 함수 구현 (enum MSG,void sendMSG()  void recvMSG() | 졸작회의 메세지 식별 함수 구현 (enum MSG,void sendMSG()  void recvMSG() | Team Meeting | 플레이어 아이디 부여&사망 void IsPlayerDead();  **void SC\_sendPlayerData();** ->id 포함 | 졸작회의 플레이어 아이디 부여&사망 void IsPlayerDead();  void SC\_sendPlayerData(); ->id 포함 | 플레이어와 벽/총알충돌처리 -> 서버 |
| 11/12 | 11/13 | 11/14 | 11/15 | 11/16 | 11/17 | 11/18 |
|  | time송수신 SC\_sendServerTime(); | 졸작회의  중간고사 time송수신 void SC\_recvServerTime(); | Team Meeting | 코드 정리 |  |  |
| 11/19 | 11/20 | 11/21 | 11/22 | 11/23 | 11/24 | 11/25 |
| 플레이어 관전 구현 | 플레이어 관전 구현 | 졸작회의 플레이어 관전 구현 | Team Meeting | 여유되면 채팅서버 | 졸작회의 |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **일** | **월** | **화** | **수** | **목** | **금** | **토** |
| 11/26 | 11/27 | 11/28 | 11/29 | 11/30 | 12/1 | 12/2 |
|  |  |  | Team Meeting |  | 테스트 및 마무리 | 테스트 및 마무리 |
| 12/3 |  |  |  |  |  |  |
| 테스트 및 마무리 |  |  |  |  |  |  |