네트워크 게임 프로그래밍 (01)

Project Progress Report

3팀

2020184015 박가현

2020184025 이승희

2020184032 정다민

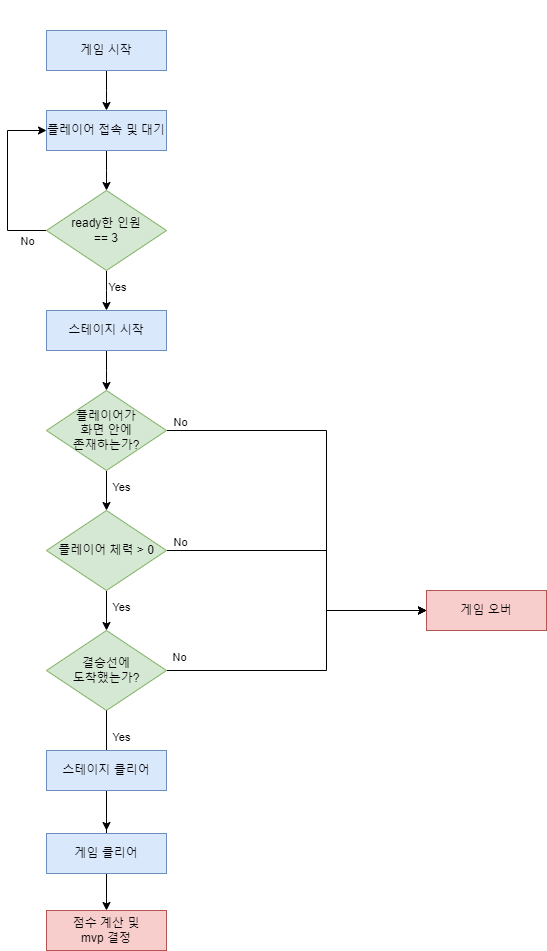
목차

1. **프로그램 개요**
2. **게임의 특징**
3. 게임 진행
4. 장애물
5. 아이템
6. 조작법
7. **개발 환경**
8. **High Level Design**
9. 클라이언트
10. 서버
11. **Low Level Design**
12. 클라이언트 + 서버
13. 클라이언트
14. 서버
15. **역할 분담 및 개발 일정**
16. **프로그램 개요**

|  |  |
| --- | --- |
| 게임명 | To Be a Butterfly |
| 교과목 | 윈도우프로그래밍 |
| 기존 클라이언트 작업자 | 2020184025 이승희 |
| 장르 | 캐주얼 아케이드 게임 |
| 플레이어 수 | 3명 |

|  |  |
| --- | --- |
|  |  |
|  |  |

1. **게임의 특징**

****

1. 게임 진행

- 게임의 스테이지는 총 1개이고, 캐릭터는 총 3개이다.

- 캐릭터의 체력은 150이다.

- 화면에 본인의 체력 UI와 팀원들의 체력 UI가 존재한다.

- 스크롤이 자동으로 진행되고, 캐릭터가 화면 아래로 벗어나면 게임 오버된다.

- 스테이지 클리어 후에 동전을 가장 많이 모은 플레이어는 MVP가 된다.

- 이벤트를 사용하여 스레드 동기화를 지원한다.

1. 장애물
2. 일반 구름

- 스테이지를 이루는 기본 장애물이다.

- 캐릭터가 통과할 수 있지만, 그동안 체력과 속도가 감소한다.

1. 먹구름

- 구름의 일정 범위 아래까지 비를 내린다.

- 캐릭터가 비를 맞으면 체력이 감소한다.

- 캐릭터가 비를 맞으면 일정 시간동안 캐릭터가 멈춘다.

1. 번개 구름

- 구름의 일정 범위 아래까지 번개를 내리친다.

- 캐릭터가 번개에 맞으면 체력이 감소한다.

- 캐릭터가 번개에 맞으면 일정 시간동안 캐릭터가 멈춘다.

1. 아이템
2. 동전

- 스테이지 클리어 시 동전을 가장 많이 획득한 플레이어가 MVP가 된다.

1. 하트

- 획득 시 체력이 15 회복된다.

1. 조작법

- 키보드의 방향키를 이용하여 상, 하, 좌, 우로 이동한다.

1. **개발 환경**

- 운영체제: Windows 10

- 통신 프로토콜: TCP/IP

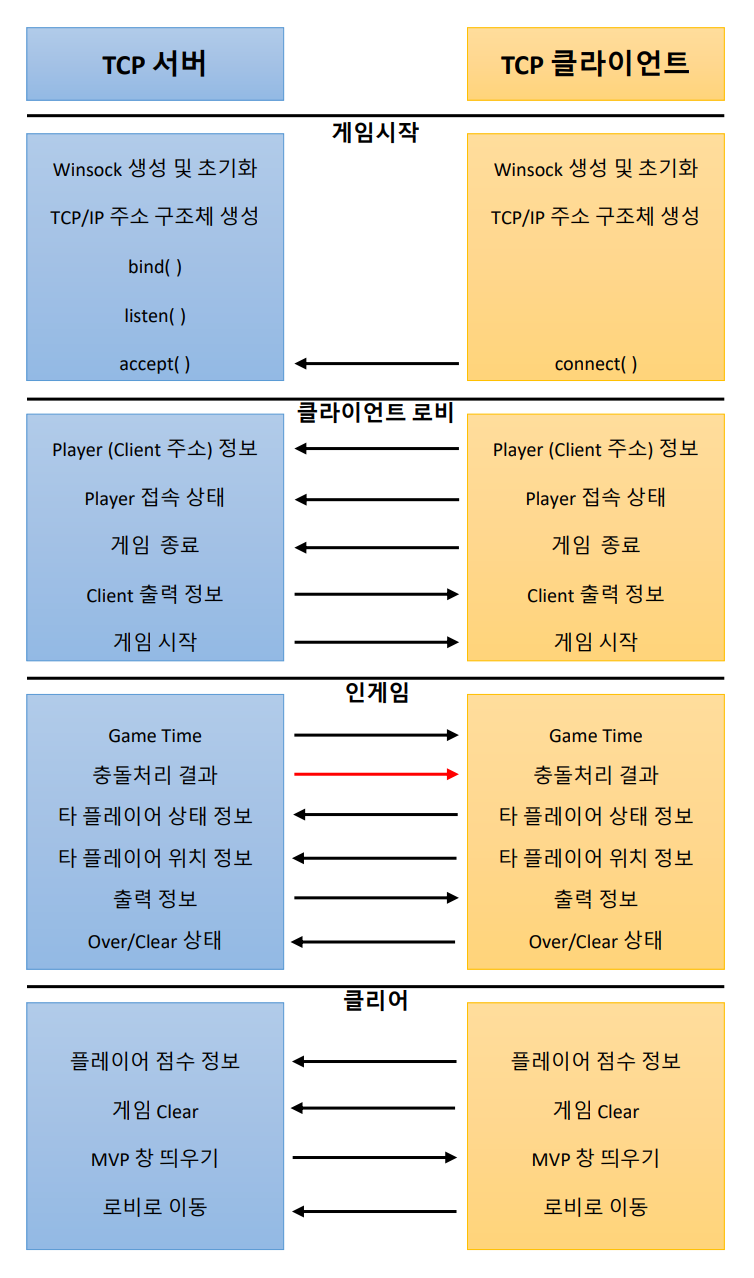
- 사용 언어: C/C++

- 컴파일러: Visual Studio 2022

- 라이브러리: OpenGL

- 버전 관리: github

1. **High Level Design**

(서버 – 클라이언트 흐름도)

|  |  |
| --- | --- |
| 서버 | 클라이언트 |
|  |  |

1. 클라이언트

- 프로그램 실행 후 서버에 접속한다.

- 이벤트를 이용하여 접속 인원이 3명이 될 때까지 기다린다.

- 게임 진행 중에는 서버로부터 게임 데이터와 카메라 시점 정보를 받아 게임 화면을 렌더링한다.

- 플레이어의 아이템 충돌 및 상태 정보를 서버로 보낸다.

- 게임 오버 시 게임 오버 창을 렌더링하고, 다른 클라이언트들이 게임을 클리어할 때까지 대기한다.

- 게임 클리어 시 게임 클리어 창을 렌더링하고, 다른 클라이언트들이 게임을 클리어할 때까지 대기한다.

- 모든 클라이언트가 게임 클리어 시 MVP 결과를 받아 결과 창을 렌더링한다.

- 프로그램을 종료한다.

1. 서버
2. 서버 실행

- WinSock을 초기화하고 대기 소켓을 생성하고 bind 함수와 listen 함수를 통해 지역 IP 주소와 지역 포트 번호를 결정하여 클라이언트의 접속을 기다리는 대기 소켓 상태로 만든다.

- 클라이언트가 connect 함수로 서버에 접근하면 accept된 클라이언트를 위한 소켓을 생성하고 로비 화면으로 전환하는 메시지 정보를 송신한다.

1. 플레이어 별 스레드를 이용한 데이터 통신

- 프로그램을 실행한다.

- 프로그램이 시작되면 event를 비신호 상태로 생성한다. 접근 순서가 중요해 이벤트를 사용하여 동기화를 진행한다.

- 접속한 플레이어 별로 스레드를 할당한다.

- 이벤트를 이용하여 3개의 클라이언트가 접속할 때까지 대기한다.

- 하트 또는 동전과 충돌이 있을 때 이벤트 비신호 상태를 신호 상태로 바꾼다.

- 하트, 동전 아이템의 생성과 삭제는 임계 영역으로 관리한다.

- 이벤트 신호 상태를 확인하여 신호 상태일 때 게임 데이터를 갱신한다.

- 통신 소켓을 생성하고 스레드 내부에서 IP 주소 정보를 받는다.

- 스레드 내부에서 클라이언트가 보낸 플레이어의 아이템 충돌 및 상태 정보를 수신 받는다.

- 각각의 클라이언트들에게 다른 클라이언트의 위치 정보, 상태 정보를 지속적으로 송신한다.

- 각각의 클라이언트들에게 카메라 시점 정보를 지속적으로 송신한다.

- 게임 종료 혹은 클리어 조건에 부합하는 순간 다른 클라이언트에게 갱신된 상태를 전송한다.

- 게임 종료 시에는 게임 종료 신호를 클라이언트에 전송하여 결과 화면으로 넘어간다.

- 결과 화면에서는 MVP 플레이어 정보를 클라이언트에 보내 결과를 출력할 수 있게 한다.

- 이벤트가 필요하지 않으면 CloseHandle() 함수를 호출하여 이벤트를 제거한다.

- 프로그램 종료 메시지를 받게 되면 클라이언트에 해당하는 스레드의 소켓을 종료하고 스레드를 제거한다.

1. **Low Level Design**
2. 클라이언트 + 서버

|  |  |
| --- | --- |
| struct tagPlayerInit | POS tPos[CLIENT\_COUNT],  TEAMNUM::TEAM team[CLIENT\_COUNT],  bool start,  int iCount,  int idx |
| struct tagPlayerInitSend | POS tPos,  TEAMNUM::TEAM team[CLIENT\_COUNT],  bool start,  int iCount = 0,  int idx |
| struct tagPlayerInfo | POS tPos,  FRAME\_INFO tFrameInfo,  int iHp,  TEAMNUM::TEAM team,  bool start,  bool isHit = false;  bool isDead = false;  ENDING::END\_TYPE eEnding = ENDING::ING;  int coinNum; |
| struct tagStoreData | PLAYER\_INFO tPlayersInfo[CLIENT\_COUNT];  int iClientIndex;  int iHp[CLIENT\_COUNT];  TEAMNUM::TEAM team[CLIENT\_COUNT];  bool start; |
| struct HeartCreate,  struct CoinCreate | int cnt,  int index,  bool bCreateOn,  POS pos |
| struct HeartDelete,  struct CoinDelete | int cnt,  int index,  bool bDeleteOn |
| struct HeartInfo,  struct CoinInfo | \_\_Create t\_\_Create,  \_\_Delete t\_\_Delete |
| struct Cloud | POS pos;  int type; |
| struct tagHeartRes,  struct tagCoinRes,  struct tagCloudRes | bool bCollision,  int iIndex |

1. 클라이언트

|  |  |  |
| --- | --- | --- |
| DWORD WINAPI | ServerProcess() | 서버 스레드 함수 |
| HANDLE | hGameEvent | 게임 시작 관련 이벤트 |
| HANDLE | hSocketEvent | 게임 갱신 관련 이벤트 |
| bool | RecvPlayerInit(SOCKET) | 플레이어 초기 정보 수신 |
| bool | SendRecvPlayerInfo(SOCKET) | 플레이어 정보 송수신 |
| bool | RecvCameraData(SOCKET) | 카메라 시점 정보 수신 |
| bool | SendRecvHeartInfo(SOCKET),  SendRecvCoinInfo(SOCKET) | 하트, 코인 정보 송수신 |
| void | AddHeart(HeartCreate),  AddCoin(CoinCreate),  DeleteHeart(HeartDelete), DeleteCoin(CoinCreate) | 하트, 코인 생성 및 삭제 |
| void | CollisionHeart  (list<CObj\*>&, list<CObj\*>&),  CollisionCoin  (list<CObj\*>&, list<CObj\*>&),  CollisionCloud  (list<CObj\*>&, list<CObj\*>&), | 하트, 코인, 구름 충돌 정보 갱신 |

1. 서버

|  |  |  |
| --- | --- | --- |
| DWORD WINAPI | ServerMain() | 서버 메인 스레드 함수 |
| DWORD WINAPI | ClientThread() | 각 클라이언트와 통신을 담당할 스레드 함수 |
| HANDLE | g\_hClientEvent[CLIENT\_COUNT] | 각 클라이언트 게임 갱신 관련 이벤트 |
| void | InitItem(),  InitCloud() | 하트, 코인, 구름 정보를 받아오는 함수 |
| bool | SendPlayerInit() | 플레이어 초기 정보 송신 |
| bool | SendRecvPlayerInfo() | 플레이어 정보 송수신 |
| bool | SendCameraData() | 카메라 시점 정보 송신 |
| bool | SendRecvHeartInfo(),  SendRecvCoinInfo(),  SendCloudInfo(), | 하트, 코인 정보 송수신,  구름 정보 송신 |

1. 역할 분담 및 개발 일정



