네트워크 게임 프로그래밍
Term Project
-최종 보고서-

2012180053 김 홍 주
2012180061 임 호 혁

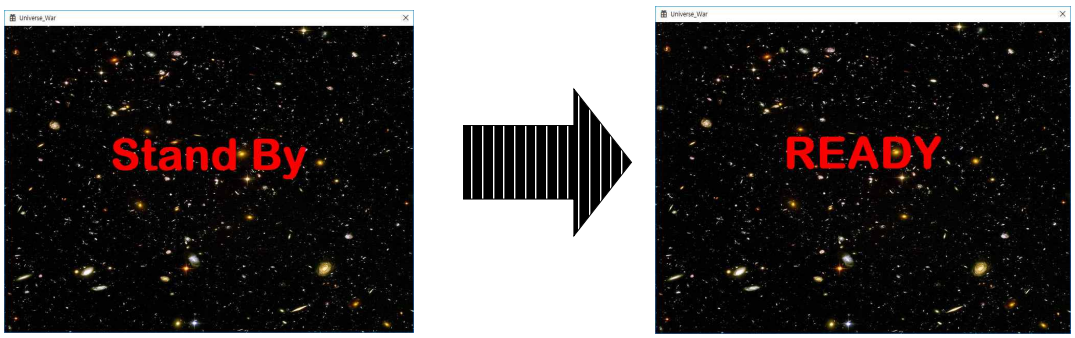
애플리케이션 기획

- 게임 제목 : **Universe War**

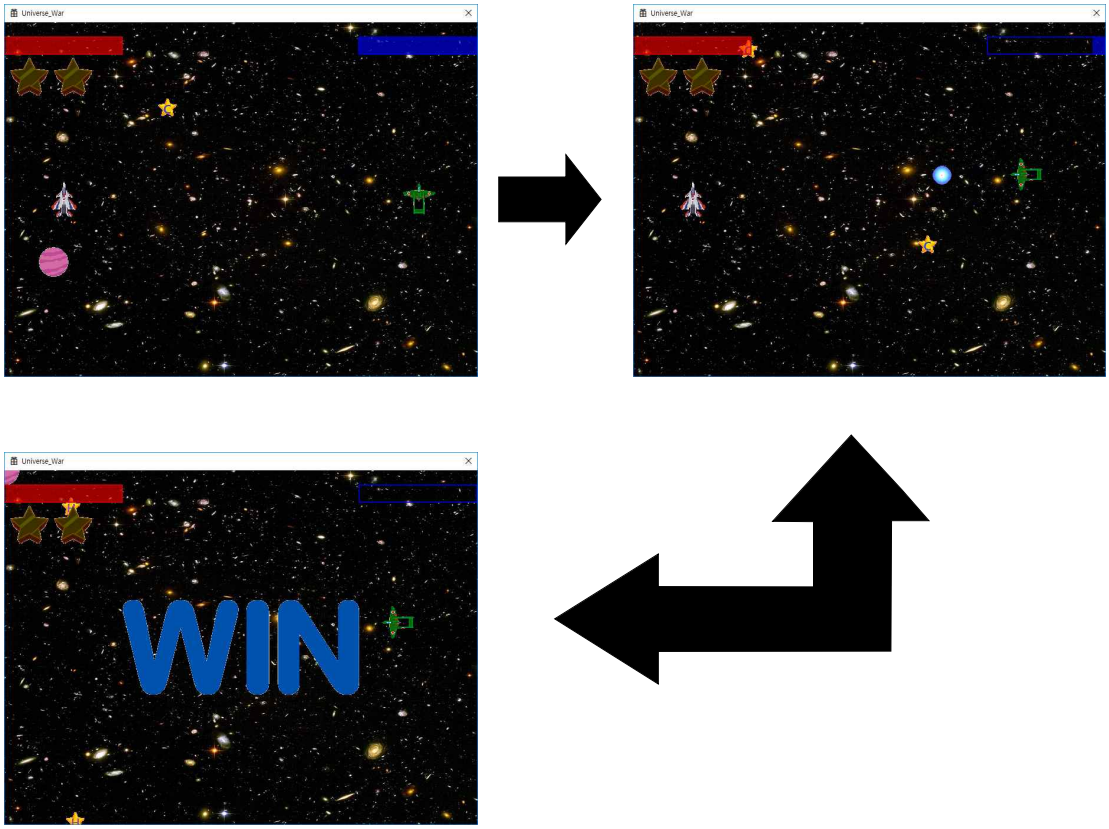
- 게임 설명 : 네트워크 통신을 하여 2인 대결 플레이를 할 수 있다. 총알 피하기 게임에 추가적 기능을 넣어 행성들을 피하면서 각 플레이어들끼리 대결하는 방식이다. 두 명의 플레이어 중에 한 명만 남게 되면 게임은 종료된다.

- 제작한 게임

● ReadyScene



● GameScene



- 게임 플레이 방식

● 800 x 600 크기로 한다.
● 1vs1 대결형식 게임이다.
● 대기하는 공간에서 각 클라이언트들이 모여 준비 상태를 만든다.
● 게임 시작 시 플레이어들은 각 위치에서 시작한다.
● 사방에서 행성 오브젝트들이 나타난다.
● 행성들과 플레이어간의 충돌 시 체력이 감소한다.
● 플레이어들은 미사일을 사용하여 공격을 할 수 있다.
● 미사일은 행성들과 충돌 시 미사일만 제거 된다.
● 다른 플레이어와 미사일 충돌 시 체력이 감소한다.
● 시간이 지날수록 행성의 움직임 속도가 빨라진다.
● 아이템 종류는 2가지가 있다.
    1. 각 플레이어의 체력을 바꾸는 아이템
    2. 플레이어의 체력을 보충하는 아이템
● 플레이어 중 한명이 죽으면 게임은 끝난다.

- **조작 방법**
  플레이어 방향키 : ←,→,↑,↓
  공격키 : space bar
  아이템 사용키 : z

- Server구현

● 클라이언트가 접속을 했는지 확인한다.
● 클라이언트가 대기인지 준비인지 확인한다. (게임 시작 전)
● 게임 시작 시 클라이언트에게 타이머와 각 클라이언트 화면 위치를 정해준다.
● 행성 및 아이템의 생성 및 위치 갱신을 시켜준다.
● 각 객체들의 충돌판별을 해준다.
● 각 객체들을 관리해준다.
● 게임이 끝나면 클라이언트들을 다시 대기 상태로 만들고 Scene을 바꿔준다.

- Client구현

● 서버로 연결하였으면 대기상태 또는 준비상태를 만든다.
● 각 객체들을 업데이트 및 렌더링을 해준다.
● 서버에서 받은 데이터들을 적용해준다.
● 플레이어가 미사일 공격을 하면 서버로 공격 정보를 보내준다.
● 플레이어가 아이템을 사용 시 서버로 아이템 사용정보를 보내준다.

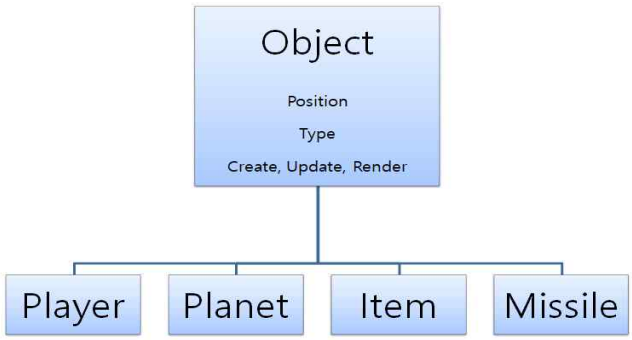
High-level 디자인

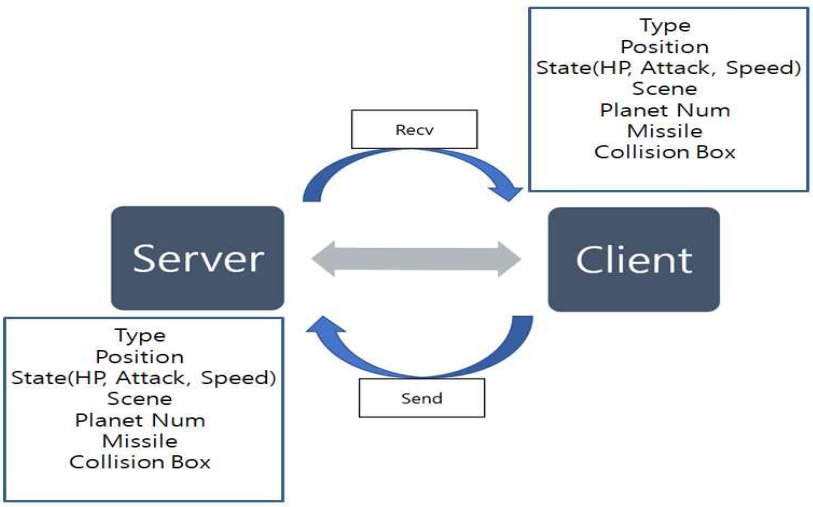
Server

- 두 명의 플레이어가 모두 connect를 하면 서버는 접속한 클라이언트에게 받아 다른 클라이언트로 초기화 위치, 총알, 아이템의 정보를 전송한다.
- 플레이어들의 정보를 받아서 서로의 충돌처리, 아이템, 행성과의 충돌처리를 한다.
- HP바꾸기 아이템 사용 시 받은 클라이언트의 ID의 HP값을 서로 변경해준다.
- HP회복 아이템 사용 시 획득한 클라이언트의 ID를 구별해 HP를 회복시킨다.
- 두 명의 플레이어중 한 플레이어가 살아남으면 두 명의 플레이어에게 한명은 WIN, 한명은 LOSE결과를 보내준다.
- 1초에 30번 연산을 한 후, 클라이언트에게 전송한다.

Client

- 게임 프레임워크를 만든다.
- 각 오브젝트들을 클래스화 시킨다.
- 서버로부터 오브젝트들의 Update State를 받는다.





Low-level 디자인

공용

◆ float Time; // 게임 진행 시간을 서버로부터 받는다.

◆ enum GAMESTATE{ WAIT, READY, GAMING };

- 게임 진행 상황 전달 값

◆ enum KEYBOARD { RIGHT, LEFT, UP, DOWN };

- 플레이어 입력 키 값

◆ enum CLIENTSCENE { SCENE1, SCENE2 };

- 클라이언트들의 화면 위치

◆ enum ATTACKSTATE { ATTACK, NON_ATTACK };

- 공격에 대한 상태

◆ enum FIGHTSTATE { WIN, LOSE };

- 클라이언트 승 패 상태

◆ enum OBJ_TYPE { PLAYER, ITEM, PLANET, MISSILE };

- 서버와 클라이언트에게 정보를 넘길 때 필요한 정보

◆ enum ITEM_TYPE{ ITEM1, ITEM2, NON_ITEM };

- 아이템 종류

◆ struct Point{ float x, float y};

- 플레이어, 아이템, 행성, 미사일의 위치 값

Server

- DWORD WINAPI CreatePlayerThread(LPVOID arg);
  - 멀티스레드를 이용하여 각 클라이언트들 정보들을 송수신 역할을 한다.

- void init() – 플레이어들의 화면 위치, 미사일, 행성, 아이템들 초기화
- bool ReadyAllPlayer(State, Player) - 플레이어1, 2가 모두 준비가 되면 게임이 시작 된다
- void Collision() – 플레이어와 미사일, 플레이어와 아이템 및 미사일과 행성, 행성과 플레이어 충돌 처리하는 함수
- void Result(Time, Player, Result) –플레이어에게 win or lose 결과 값과 시간을 보내주는 함수
- void SendInfo(Missile, Planet, Item, Player) – 플레이어들에게 아이템과 미사일, 행성 정보를 보내주는 함수
- void CheckKeyboard() – 키보드 입력 처리 함수
- void Operation_ProcessThread() – 클라이언트에서 넘어온 정보 연산처리 스레드 만드는 함수
- void GetHpItem(Player_hp) – 플레이어가 HP회복 아이템을 먹으면 체력을 올려주는 함수
- void ChangeHpBar(Player_hp, Player_hp) – 아이템 사용 시 플레이어 2명의 hp를 바꿔주는 함수
- void CreateItems(Item_type, Player) - 아이템을 생성해서 획득한 플레이어에게 효과를 발휘할 수 있게 정보(HP변경, HP회복)를 보내주는 함수

Client

```
class CPlayer
{
	OBJ_TYPE type;
	CLIENTSCENE scene;			// 플레이어의 화면 위치
	Point pos;				// 플레이어의 위치 값
	float hp;				// 플레이어의 체력
	bool GameState;				// 게임의 상태
	CMissile* pMissile[MAXMISSILE];		// 미사일의 정보
	CITEM* pItem[MAXITEM];			// 아이템의 정보
}

class CPlanet
{
	OBJ_TYPE type;
	CLIENTSCENE scene;			// 행성 화면의 위치
	Point pos;				// 행성 위치 값
}

class CItem
{
	OBJ_TYPE type;
	Point pos;
	ITEM_TYPE itemType;
	// 아이템 타입을 정하는 변수이므로 switch문 사용할 예정
}
```

```
class CMissile
{
	OBJ_TYPE type;
	Point pos;
	float attack;
}
```

객체들의 공용 함수

void Create(HWND hWnd) - 객체들을 초기화 시킨다.

void Update(float time) - 객체들을 갱신해준다.

void Render() - 객체들을 그려준다.

함수 선언

void GameState(bool state);

- 서버에서 클라이언트들이 준비 상태이면 변경해준다.

bool CollisionCheck(object, object);

- 충돌체크

void AttackKey()

- 공격 키를 누르게 되면 서버쪽으로 명령 패킷을 보낸다.

void ChangeHP()

- 서버에서 플레이어들의 체력을 바꿔주는 정보를 받았는지 판별

void sendObj()

- 객체들의 정보들을 서버로 보내준다.

void recvObj()

- 객체들의 정보를 받아 업데이트 해준다.

개발환경

운영체제	Windows 10
플랫폼	Visual Studio 2017 Window API (C/C++)
네트워크	TCP / IP 멀티스레드

역할분담

김 홍 주	임 호 혁
서버 프레임워크 제작	클라이언트 프레임워크 제작
서버 -> 클라이언트 GameScene제작	서버 -> 클라이언트 ReadyScene제작
서버 -> 클라이언트 Object제작	서버 -> 클라이언트 Object제작
서버 송수신, 클라이언트 송수신 제작	서버 송수신, 클라이언트 송수신 제작