**추진 계획서**

**네트워크 게임 프로그래밍**



|  |  |
| --- | --- |
| 담당교수 | 김재경 교수님 |
| 학생 | 정휘현 (2013180036) |
| 김동현 (2013182007) |
| 이용선 (2013182034) |

**1. 애플리케이션 기획**

|  |  |
| --- | --- |
| 게임 제목 | Bomb |
| 게임 장르 | Arcade |
| 플랫폼 | PC |

▶ **게임 기본정보**

표 1 - 게임 기본정보

▶ **게임 아이디어**

그림 2 봄버맨



- 기본적인 게임의 형태는 봄버맨 게임의 토대를 그대로 사용하여 진행하였습니다.

- 게임을 제작해야 하는 상황이기 때문에 텍스쳐를 제외하고 제작하기로 생각하였으며 각 게임의 오브젝트들을 사각형, 또는 원을 통해 표현하기로 하였습니다. (색상으로 구별)

▶ **게임 설정**

**- 맵**

|  |  |  |
| --- | --- | --- |
| **오브젝트** | **사용 컨테이너** | **개수** |
| 맵 | 배열 | 12 x 12 |

**- 구분 색상**

|  |  |
| --- | --- |
| **오브젝트** | **색상** |
| 플레이어 1P | Red |
| 플레이어 2P | Blue |
| 플레이어 3P | Green |
| 파괴되는 벽 | Texture (wood) |
| 파괴되지 않는 벽 | Texture (block) |
| 폭탄 | Texture (bomb) |

**- 폭탄**

|  |  |  |
| --- | --- | --- |
| **오브젝트** | **컨테이너** | **개수** |
| 폭탄 | list | 3 |

|  |  |
| --- | --- |
| 폭탄 범위 | 3x3 |

**2. 개발환경**

|  |  |
| --- | --- |
| OS | Windows |
| 개발 툴 | 비쥬얼 스튜디오2017 |
| 개발 언어 | C++ |
| 사용 라이브러리 | DirectX2D, WINAPI |
| 플레이 인원 | 3인 |

**3. High-Level 디자인**

그림 3 서버 디자인

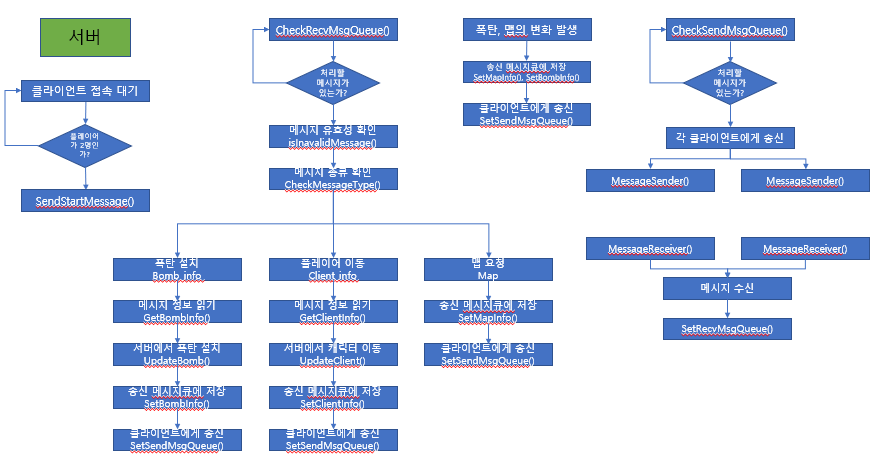
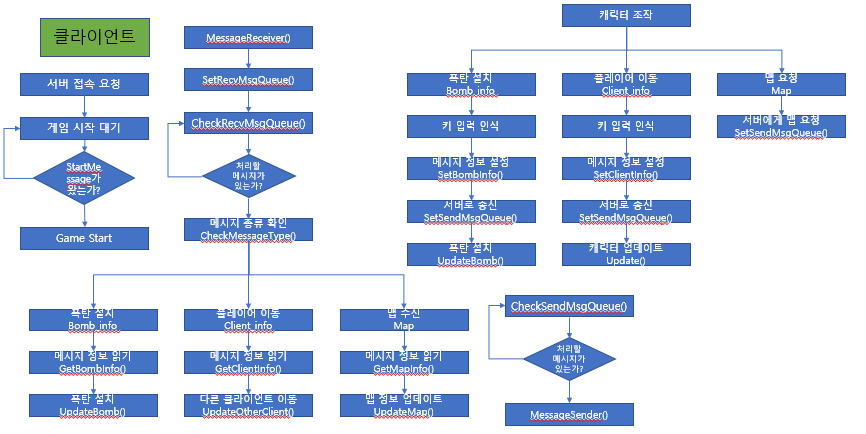
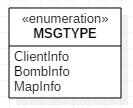


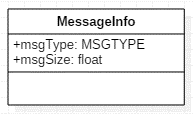
그림 4 클라이언트 디자인



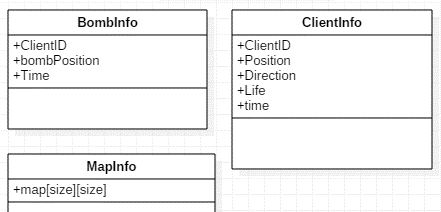
**4. Low-Level 디자인**



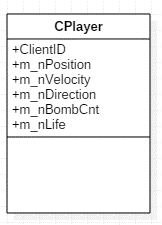
고정-가변 데이터 전송 시 앞으로 받아야할 데이터가 어떠한 데이터의 타입인지에 대한 Enum 데이터



고정 데이터(앞으로 받을 데이터의 사이즈 전송) 구조체

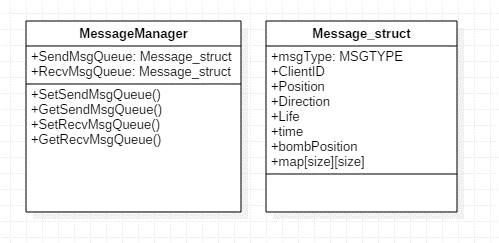


각 정보를 저장하고 전달하는데 사용할 구조체 데이터



플레이어 클래스

MessageManager 클래스



|  |  |
| --- | --- |
| **함수 목록** | **내용** |
| DecodingMsg(MessageInfo msg) | MessageInfo로 넘겨받은 데이터들을 클라이언트 내부 객체들과  동기화 시키는 작업 |
| MessageInfo PackageKeyInput() | 입력된 키 값 중  서버로 보내야 할 데이터들을  MessageInfo로 패키징 하는 함수 |
| (서버)DWORD WINAPI MessageSender() | 서버가 전체 클라이언트  대상으로 메시지 전송 |
| (서버)DWORD WINAPI MessageReceiver() | 클라이언트가 서버에서 보내는 메시지 수신 |
| (클라)DWORD WINAPI MessageSender() | 클라이언트가 서버에게  요청하는 메시지 전송 |
| (클라)DWORD WINAPI MessageReceiver() | 서버가 클라이언트 요청 메시지 수신 |
| bool CheckRecvMsgQueue()  bool CheckSendMsgQueue() | RecvMsgQueue나 SendMsgQueue에 메시지가 들어있는지 확인하는 함수 |
| DWORD timeGetTime() | 서버의 시간을 구하는 함수 |
| Message\_struct GetSendMsgQueue()  Message\_struct GetRecvMsgQueue() | MsgQueue에서 첫 번째 메시지를 pop하는 함수 |
| void SetSendMsgQueue()  void SetRecvMsgQueue() | MsgQueue에 서버나 클라이언트에서 처리한 사항이나 요청할 메시지를 담는 함수 |
| bool isInvalidMessage() | 서버에서 메시지의 내용, 시간을 고려하여 유효한 메시지인지 판단하는 함수 |
| int CheckMessageType() | 메시지의 종류를 판단하는 함수 |
| void SetXXXInfo() | XXX에 해당하는 정보를 Message\_struct에 저장하여 바로 MsgQueue에 저장할 수 있도록 준비하는 함수 |
| Message\_struct GetXXXInfo() | XXX에 해당하는 정보를 MsgQueue에서 가져오는 함수 |
| void UpdateXXX() | XXX에 해당하는 정보를 서버나 클라이언트에서 정보갱신하는 함수  전체 Update()함수와는 달리 그 정보만을 갱신한다. |
| DWORD WINAPI WaitingForSingleObject() | MsgQueue가 공유자원이므로 SendMessage나 RecvMessage, GetMsg, SetMsg등 함수들이 동시에 접근하지 못하도록 제어하는 함수 |
| void SendStartMessage() | 접속하고 대기 상태인 클라이언트들에게 게임시작을 알리는 함수 |
| void SendEndMessage() | 게임이 종료되는 조건이 만족됐을 때 클라이언트들에게 게임종료를 알리는 함수 |

**5. 팀원 별 역할 분담**

|  |  |  |
| --- | --- | --- |
| **이름** | **주 업무** | **보조 업무** |
| 정휘현 | 클라이언트 | 서버 |
| 김동현 | 서버 | 서버 |
| 이용선 | 서버 - 클라이언트 | 클라이언트 |

**6. 개발 일정**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **직군** | **이름** | **1주차 (11/11 토 ~ 17 금)** | | | | | | |
| **11/11** | **11/12** | **11/13** | **11/14** | **11/15** | **11/16** | **11/17** |
| 클라 | 정휘현 |  |  |  |  |  | 2d게임 프레임  워크 제작 | 2d 리소스 검색 |
| 서버 | 김동현 |  |  |  |  |  | MessageManager 구현 | MessageManager 구현 |
| 서버 | 이용선 | Game  Object  Class  구현 | 대전 (결혼식장) | 통신 테스트를 위한 프로토타입 클라이언트 제작 | 프로토타입 클라이언트  (플레이어, MssageInfo 데이터) |  | MessageInfo PackageKeyInput | Message Manager를 통한 통신 테스트 |
| 클라 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 직군 | 이름 | 2주차 (11/18 토 ~ 24 금) | | | | | | |
| 11/18 | 11/19 | 11/20 | 11/21 | 11/22 | 11/23 | 11/24 |
| 클라 | 정휘현 | 지방으로 내려가 작업 일시 정지 | | 게임 오브젝트 그리기 | 1인용 게임제작 | 플레이어 추가 | 플레이어 난입과정을 싱글에서 구현 | 클라이언트 버그 검사 |
| 서버 | 김동현 | MessageSender, MessageReceiver구현 | MessageSender, MessageReceiver구현 |  | 이벤트를 이용하여 동시성제어 구현 | 이벤트를 이용하여 동시성제어 구현 |  | isInvalidMessage, CheckMsgQueue구현 |
| 서버 | 이용선 | Message Manager를 통한 통신 테스트 | 버그 확인 및 수정 |  | 공연  리허설 | 공연 |  |  |
| 클라 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **직군** | **이름** | **3주차 (11/25 토 ~ 12/1 금)** | | | | | | |
| **11/25** | **11/26** | **11/27** | **11/28** | **11/29** | **11/30** | **12/1** |
| 클라 | 정휘현 | 프로그램 정검 | | 오브젝트 처리를 위한 헤더파일을 서버에 이식 | 키 입력에 따른 처리 함수 서버에 이식 | 충돌상태 검사와 효과 함수 서버에 이식 | 업데이트 처리함수 버그 확인 | 업데이트 처리함수 버그확인 |
| 서버 | 김동현 | 클라이언트 프레임워크 서버에 이식 | GetXXXInfo, SetXXXInfo 구현 | UpdateXXX구현, 클라이언트와 서버 통신동작 확인 | 서버와 클라이언트 동기화 확인 |  |  | 멀티 클라이언트 지원 확인 |
| 서버 | 이용선 | BroadcastSender, Message Receiver통신 테스트 | BroadcastSender, Message Receiver통신 테스트 | 통신 테스트간 버그 확인 및 수정 | 통신 테스트간 버그 확인 및 수정 |  |  |  |
| 클라 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **직군** | **이름** | **4주차 (12/2 토 ~ 12/8 금)** | | | | | | |
| **12/2** | **12/3** | **12/4** | **12/5** | **12/6** | **12/7** | **12/8** |
| 클라 | 정휘현 | 오류 수정 | 서버와 클라이언트 1:1 통신동작 확인 | 서버에서 받아온 데이터 그리기 | 서버에서 받아온 데이터 그리기 | 그리기 작업 오류가 있으면 수정 | 그리기 작업 오류가 있으면 수정 | 서버와 클라이언트 1:N 통신동작 확인 |
| 서버 | 김동현 |  |  | 멀티 클라이언트 서버간 동기화 확인 | 멀티 클라이언트 서버간 동기화 확인 |  |  | 다른 네트워크 에서 접속여부 확인 |
| 서버 | 이용선 |  |  |  |  |  |  |  |
| 클라 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **직군** | **이름** | **5주차 (12/9 토 ~ 12/14 목)** | | | | | | |
| **12/9** | **12/10** | **12/11** | **12/12** | **12/13** | **12/14** | **종강** |
| 클라 | 정휘현 | 서버에서 받아온 다른 플레이어 그리기 | 서버에서 받아온 다른 플레이어 그리기 | 그리기 작업 오류가 있으면 수정 | 그리기 작업 오류가 있으면 수정 | 최종 프로젝트 테스트 | 최종 프로젝트 제출 |
| 서버 | 김동현 | 다른 네트워크에서 접속 확인 |  | 최종 동작 확인 및 디버그 | 최종 동작 확인 및 디버그 |
| 서버 | 이용선 | 다른 네트워크의 컴퓨터와의 연결테스트 | 다른 네트워크의 컴퓨터와의 연결테스트 | 버그 확인 | 버그 확인 |
| 클라 |