네트워크 게임 프로그래밍 추진서

2016182024 윤선규

2016182026 이동수

2017180015 서동현

**애플리케이션 소개**

기획 - 점프하여 상대방을 밟고 승리하라 ( J&A Jump&Attack )

조작키 - 이동: 좌우방향키, 점프: 스페이스바

**게임진행**

플레이어들이 좌우로 이동, 점프.

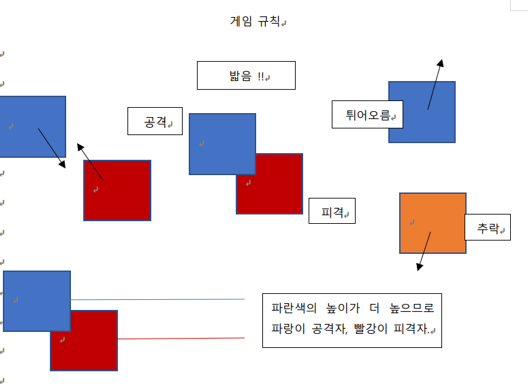
상대방을 밟으면 상대방은 체력을 소모, 색깔이 바뀐다.

체력에 따라 빨강, 주황, 노랑, 초록, 파랑, 남색, 보라 무지개 색으로 바뀌다가 체력이 0이 되면, 멈춘다. 검정색이 되어 움직일 수 없고, 스페이스바를 누르면 랜덤 한 위치에서 부활한다.

플레이어는 떨어지다가 상대방을 찍으면 한 번 더 튀어오르고 찍힌 상대가 점프를 하는 중이면 상대는 즉시 아래로 떨어진다.

이벤트 드라이븐 방식을 사용하여, 클라이언트들을 동기화 시킨다.

자세한 내용은 아래에서 설명한다.



충돌 판정은 위와 같다.

충돌판정은 AABB 충돌 시 높이 값을 기준으로 공격자와 타격자를 판정,

공격자는 한 번 더 튀어오르고, 피격자는 색이 변하며 추락한다.

이를 네트워크로 동기화 시키는 방법은 두 가지가 있다.

충돌을 검사하는 방법

1. 내 충돌만 검사해서 다른 클라이언트들에게 통보

* 8인 게임에서 나랑 나머지 7명만 계산 후 네트워킹 통보.

1. 모든 캐릭터들의 충돌을 계산

* 8인 게임에서 8\*7 56번 계산.

프로젝트를 진행하면서 두 방식 중 하나를 선택할 것이다.

**이벤트 드라이븐 방식의 동기화**

클라이언트가 서버에게 보낼 이벤트들

enum class EVENT\_TYPE

{

CREATE\_OBJECT,

DELETE\_OBJECT,

MOVE\_OBJECT,

COLLION\_OBJECT,

SCENE\_CHANGE,

END,

}

struct Event

{

EVENT\_TYPE eEvent,

DWORD\_PTR wParam,

DWORD\_PTR lParam

}

이벤트 구조체를 패킷으로 사용합니다, 모든 이벤트에서 같은 종류의 패킷을 사용합니다. eEvent를 보고 wParam 과 lParam이 어떠한 정보인지 알아낸 뒤 사용합니다.

접속

접속한 클라이언트는 서버에게 자신이 접속하였다는 메시지를 주고

랜덤하게 정해진 새로운 객체의 위치 값을 보내준다.

서버는 새로운 클라이언트에게 다른 클라이언트들의 정보를 보내준다.

다른 클라이언트들은 CreateObj 라는 함수를 호출하여 새로운 객체를 생성하고, 받은 위치 값을 적용한다.

나의 위치를 새로운 클라이언트에게 전송한다.

새로운 클라이언트는 다른 클라이언트들에게서 받은 정보로 CreateObj 함수를 호출한다.

서버의 요청에 따라 클라이언트가 호출할 함수명 :

AcceptClient(Event, Vec2); // 질문거리,, wParam과 vec2ㅁ>ㄴㅇ.asf?

움직임

클라이언트에서 플레이어가 움직이면 서버에 위치 값을 전달하여 플레이어가 움직였음을 전달해주거나 플레이어가 해당 키를 눌렀음을 전달하고

서버는 다른 클라이언트들에게 플레이어가 보낸 키 데이터나 위치 데이터를 보내고 다른 클라이언트에서 받은 정보로 직접 키 데이터를 처리해주거나 위치 데이터로 해당 플레이어를 이동시켜준다.

서버의 요청에 따라 클라이언트가 호출할 함수명 :

MoveClient(Event, Vec2); // 질문거리,, wParam과 vec2ㅁ>ㄴㅇ.asf?

접속끊김

클라이언트가 접속이 끊겼을 때 서버는 다른 클라이언트들에게 접속이 끊긴 클라이언트를 알려주어 오브젝트를 삭제한다.

함수명: DisconnectClient(Event, Vec2); // 질문거리,, wParam과 vec2ㅁ>ㄴㅇ.asf?

충돌

충돌을 검사하는 방법

1. 내 충돌만 검사해서 다른 클라이언트들에게 통보

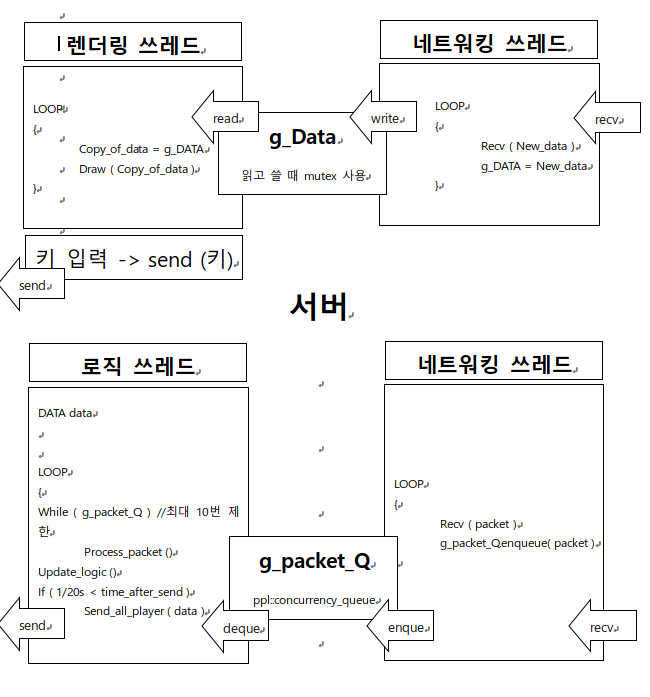
* 8인 게임에서 나랑 나머지 7명만 계산 후 네트워킹 통보.

1. 모든 캐릭터들의 충돌을 계산

* 8인 게임에서 8\*7 56번 계산.

함수명: CollisionClient(Event, Vec2); // // 질문거리,, wParam과 vec2ㅁ

클라이언트와 서버의 쓰레드 로직



**클라이언트에 구현해야 할 네트워킹 함수들**

1. Server로 Player input을 보내는 함수

2. Server에게서 Player들의 데이터들을 받는 함수

3. 받은 데이터들을 그리는 함수

**서버에 구현해야 할 네트워킹 함수들**

1, Client에게 받은 패킷을 적절한 Client들 에게 보내주는 함수

**고민해야 할 것들**

각 이벤트에 따라 전송될 정보, 패킷 구성.

패킷을 해석할 함수

충돌 계산 방법 선택

**\* 추진 일정표**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **일** | **월** | **화** | **수** | **목** | **금** | **토** |
|  | 11/08 | 11/09 | 11/10  서동현:  클라이언트 구현 아이디어 떠올리기 | 11/11  서동현:  클라이언트 뼈대  이동수:  Client 구현 | 11/12 | 11/13  이동수:  Client 구현  서동현:  클라이언트 구현 |
| 서동현:  클라이언트 구현 | 11/15  서동현:  클라이언트 구현 | 11/16 | 서동현:  작동점검 | 이동수:  RecvData구현 |  | 이동수:  RecvData구현 |
|  | 11/22 |  |  | 이동수:  DrawOtherplayer |  | 이동수:  DrawOtherplayer |