소개

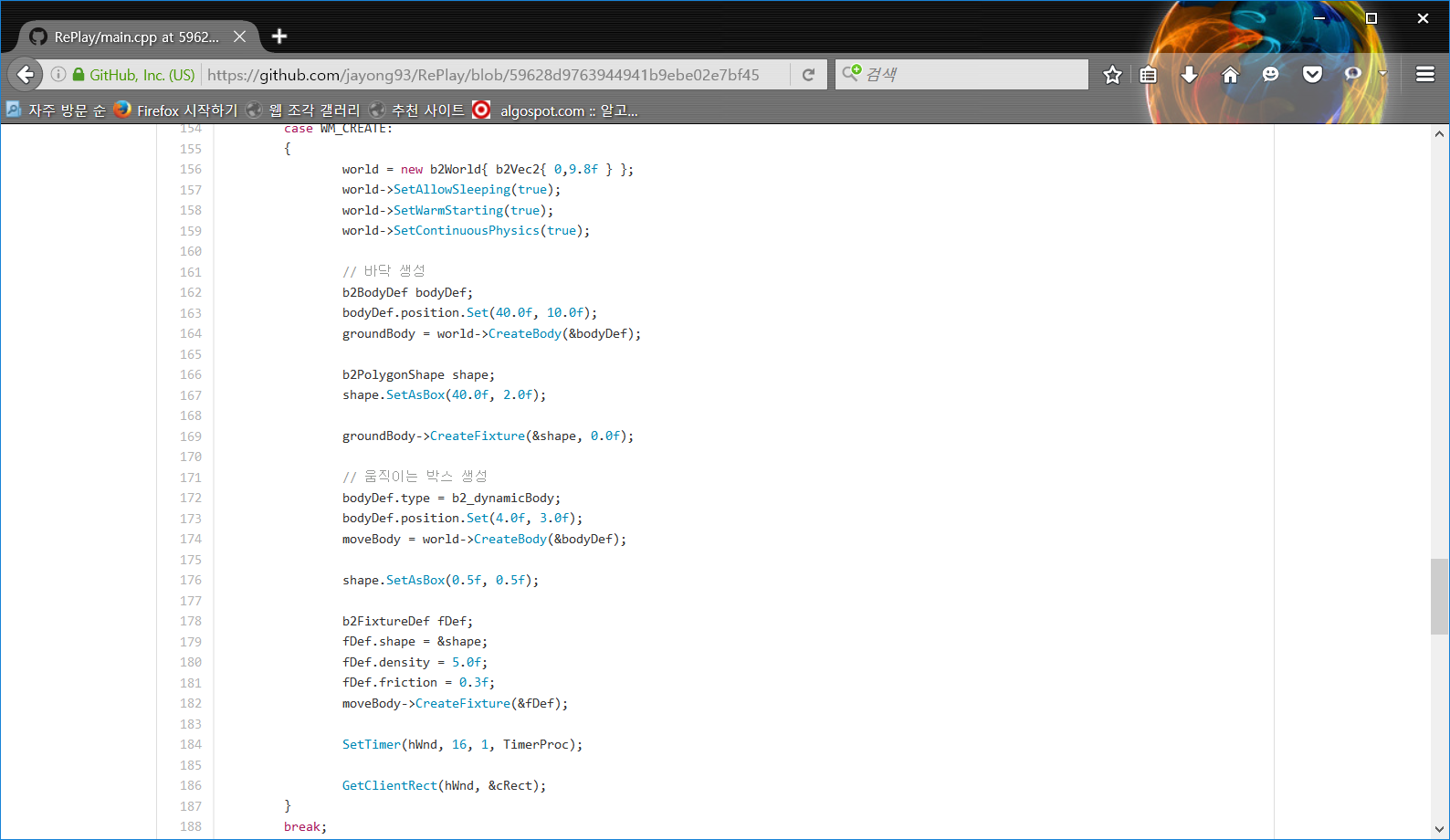
저는 이번 리플레이 과제에서 박스들을 가지고 노는 프로그램을 만들었습니다. 이 박스들은 Box2D 강체 물리엔진 라이브러리를 이용해서 물리법칙에 따라 움직이도록 구현했습니다. 제 목표는 이 복잡한 물리 계산이 들어간 움직임을 리플레이로 그대로 복구하는 것이었습니다.

처음 기획때는 당구대 위에 공들은 만들고 공을 마음대로 굴리는 느낌으로 표현하려고 했지만 원 보다는 사각형이 움직이는게 시각효과가 좋을 것 같아서 바꾸었습니다.

진행과정

기본 프로그램 구현

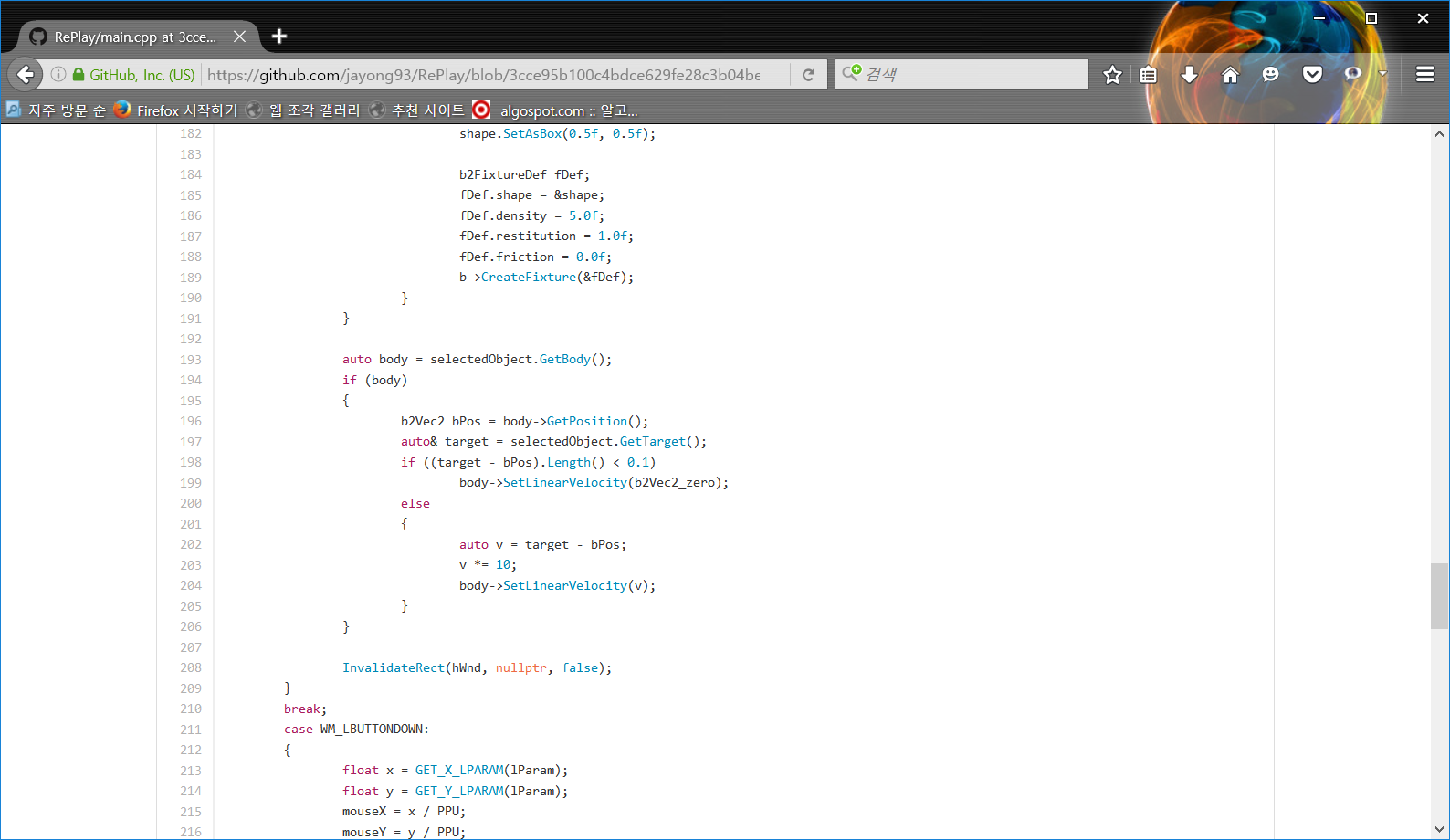
프로그램을 만들기 전에 먼저 Box2D에 대해서 공부를 했습니다. 찾아보니 Box2D는 World라는 객체를 통해서 물리 강체들을 관리하고 물리 연산을 모두 처리하는 방식으로 동작했습니다. 그래서 원래는 화면 상의 객체 모두를 STL 컨테이너로 관리하려고 했지만, 박스들은 Box2D에게 관리를 맡기고 저는 리플레이에 필요한 데이터만 STL 컨테이너로 관리하기로 했습니다.



Box2D에서 강체를 나타내는 Body는 이렇게 world 객체의 생성함수를 통해서 만들어지고 내부의 Tree 안에서 관리가 됩니다

Box2D 라이브러리가 객체를 모두 관리해주다 보니 굉장히 편하게 프로그램을 만들 수 있었습니다. 리플레이 기능을 구현하기 전까지는 제가 한 일은 사용자의 입력을 받아서 알맞은 행동(박스를 생성하거나 움직이는 것)을 연결해주는 것 밖에 없었습니다.

그렇게 해서 빈 공간에 마우스를 클릭하면 박스가 생성되고, 박스를 클릭해서 드래그하면 박스가 움직이도록 만들었습니다. 또 빈 공간에서 마우스를 드래그하면 마우스 위치에 박스를 놓았을 때 다른 박스와 충돌하는지 검사한 후에, 충돌하는 박스가 없다면 박스를 새로 생성하도록 했습니다. 이렇게 하면 선을 그리듯이 박스들을 만들 수 있습니다.



이런 식으로 박스에게 SetLinearVelocity 함수로 속도 값을 주면 물리엔진이 박스들의 상태를 계속 갱신해주고, 저는 갱신된 값들을 받아와서 계속 그려주었습니다.

리플레이 구현

기본 프로그램 구현은 Box2D의 힘 덕분에 쉽게 끝낼 수 있었지만 문제는 리플레이 기능의 구현이었습니다. 무언가를 계속 저장해 놓아야 한다는 것은 어렴풋이 알고 있었지만 정확히 무엇을 저장해야 하는지를 몰라서 꽤 오래 고민을 했습니다.