|  |  |  |  |
| --- | --- | --- | --- |
| MORPHOSIS 개발일지 17차 | | | |
| 기간 | 2019-07-02 ~ 2019-07-15 | 작성자 | 신재욱 |
| 작업 내용 | | | |
| 마우스로 입력받아서 회전하게 하는 부분에서 문제가 있었는데 입력값 없으면 키 입력이 안 되게 해놔서 그런 문제였음. 너무 급하게 짜서 그런가 이런 실수를.  마우스 관련 속도나 플레이어 이동 속도 같은 수치를 조금씩 바꿀 때마다 모든 파일 컴파일 다시 하는거 너무 짜증나는데 아예 메모장 같은걸로 밖에 빼둘까?  경사로 이동을 위해서는 선행되야 하는 것들이 있다.  일단 중력 체크를 진행해야 함. 아래 충돌체와 충돌하는 다른 충돌체가 있는지 확인?  다른 사람들은 지금 공중에 떠있는 상태인지 어떻게 체크할까? 검색해보자.    <https://solution94.tistory.com/27>  따로 지면이라는 구분을 주어서 지면과 충돌 중일 때에는 땅 위에 있다고 판정하는 방법이다.  근데 나는 지면으로 구분하려 해도 지면 충돌체와 옆으로 충돌할 수도 있기 때문에 따로 고려해주어야 할 것이 많다.    <https://truecode.tistory.com/25>  아! 그냥 충돌체의 각 아래 점에서 아래로 체크해보면 되는건가!  유니티에서의 방식을 다렉에서야 쓸 수 있겠지만 함수 등을 제공하는지의 문제는 별개이다.  점이 충돌체와 포함되는지를 알아보면 될까? 너무 빨리 이동한다면? 그럼 충돌체를 통과해버릴텐데?  <https://truecode.tistory.com/27?category=611884>  너무 빨리 이동하겠어 설마. 그렇게 빠른 이동은 내 기획엔 없다. 따라서 고려 안 해도 될 것.  저 블로그에 올라와있는 스크립트는 꽤나 쓸만해보인다. 필요한 부분만 참고해서 내 코드를 작성하자.  그 전에 다이렉트 X Math에는 레이캐스트 같은 함수가 없나 한 번 찾아보자.  없어~~~~~~~~~~  굳이 레이캐스트 기능을 만들 필요는 없지. 점이 충돌체 안에 포함되어 있으면 충돌하고 있는거 아냐.    좋아 인생 폈다. 또 바퀴살부터 만들 뻔 했네.  굳이 광선을 쏴야 할까요? 근데 광선 쏘면 뚫고 지나가는 문제를 완벽하진 않아도 조금 완화시킬 수 있지 않을까? 맞다 맞어.  일단 중력 영향 받아서 움직이게나 하자.   * 중력가속도를 저장할 변수를 만든다. * Update()에서 중력가속도 계산하고 * Move()에서 이동할 방향 계산할 때, 거기에 중력가속도만큼 아래로 더해준다. * HeightVelocity 정도로 이름을 정해야 하지 않을까? 점프를 하면 거기에 값을 더해줄텐데?   일단 코드를 짜보자!  코드를 작성하면서 CObject의 Update() 에서는 이동과 관련된 부분을 전부 삭제했다. CPlayer와 CProjectile이 각자 다른 이동방식을 갖게 되었으므로. 원래는 CObject-CMovingObject 로 이어지는 상속 구조를 가졌어야 했으나 귀찮아서 그렇게 하지 않았다. 아악.  중력 변수 자체를 갖는 것은 크게 문제 없을 것이다. 내가 이렇게 쓰레기같이 코드를 짜서 미안해 프로그램아.  하지만 생각해보면 중력의 영향은 플레이어만 받으면 되지 않을까? 투사체는 그냥 박으면 없어지면 아니 내가 무슨 생각을 투사체도 떨어지긴 해야지.  중력 영향 받게 하는 곳까진 끝. 다음은 바닥 검출이다. 다렉에는 광선을 만들어서 충돌체와 검사하는 함수가 있다. 내 바닥 충돌체의 아래 네 정점으로부터 아래 방향으로 광선을 쏘고, 그 광선에 하나라도 충돌하면  아니 근데 음……. 그냥 작은 충돌체 하나 발 밑에 만들고 그걸로 체크하면 되지 않나?  그라운드 체크 충돌체로 사용하자. 충돌하면 높이는 해당 충돌체의 위치 + extent 로 정하고 IsGround 를 true로 설정하자.  지하철을 타고 왔다 갔다하면서 책을 읽다가 책 마지막 부분에 정적 코드 분석 툴을 소개하고 있길래 사용해보았다.  <http://cppcheck.sourceforge.net/> 에서 다운로드 받아서 사용함.    헐  <https://stackoverflow.com/questions/12015156/what-is-wrong-with-checking-for-self-assignment-and-what-does-it-mean>  뭔 소리야 넘 어렵다.  충돌체와 충돌처리를 변경해야 할 것 같다. 히트박스로 사용할 충돌체와 그라운드 체크용으로 사용할 충돌체를 구분해서 사용해야 한다. 현재 사용하고 있는 방식은 다음과 같다.   * CObjectManager::LateUpdate()에서 충돌검사를 진행한다. 충돌검사는 다음과 같다.   + 플레이어가 무언가와 충돌할 수 있는 상태인지 체크하고( 그게 Enable 상태인지, 다른 팀인지)   + 실제로 충돌하는지 확인하고(충돌체 판정)   + 실제로 충돌하면 충돌정보에 대상 오브젝트 주소를 추가한다. * CPlayer::LateUpdate()에서 충돌정보에 따라 처리한다. 처리는 다음과 같다.   + Prop과 충돌했을 경우, 이동을 무효화한다.   + Player와 충돌했을 경우, 이동을 무효화한다.   지금은 내 충돌체들이 무엇과 충돌했는지 확인하여 처리하고 있다. 추가해야 할 점은 내 ‘어떤’ 충돌체가 무엇과 충돌했는지 알아야 한다. 어떻게 구현하지?  지금은 충돌체들을 총괄해서 관리하고 있음. 생각해보니까 굳이 박스랑 구랑 따로 관리해야 해? 암튼 지면충돌용 충돌체랑 다른 플레이어와 충돌, 히트 판정 용으로 쓸 충돌체를 따로 만들어서 쓰자.  하려던건 지면 충돌 검사였는데 충돌체 통합관리로 새더니 회전 시 Offset 관리까지 새버림. 집중하자…  어쨌든 지면 충돌용이 있으면 되는거 아냐?  이제와서 막 지면충돌체~ 이런 식으로 추가하면 골치아파지니까 충돌체에 태그를 넣자.  충돌체들끼리 충돌을 했는데 누가 충돌했었는지 알 수 있는 방법이 있을까?  아 완전 제대로 설계 미스 했어  지형 충돌체크는 BoundingSphere로 한다. 지형 충돌체는 전부 BoundingBox 이므로(Prop도) 그것만 기준으로 하면 된다. 충돌 여부가 확실해지면 Sphere의 중점에서 Box의 회전에 따라 정해지는 XY, YZ, XZ 평면으로 수직선을 만들고 그 평면과 수직선의 교점 A와 Box의 중점 B의 거리를 Box의 Extent + Sphere의 Radius 값과 비교하여 그보다 작으면 해당 평면 방향으로 충돌하고 있는 것이다. 여기까지 확인되면 진행방향 벡터를 해당 평면으로 투영하여 생성된 벡터가 새로운 진행방향 벡터가 된다.  이를 위해서는 지금 충돌검사를 한 뒤 이동하는 방식이 아니라, 이동할 방향으로 미리 진행한 것을 가정하고 주변 충돌체와 검사한 뒤 이동방향을 수정하고 더 이상 충돌하는 충돌체가 없을 때까지 이를 반복해야 한다.  플레이어는 진행방향이 전후좌우 상단밖에 없다. 아래로는 이동할 수 있는 방법이 없다. 그렇다면 미리 이동방향으로 검사하는 방식을 사용할 수 없다. 따라서 중력 가속도를 매 프레임마다 적용한다. 아래로 강제로 이동시키게 한다. 무언가와 충돌했고 충돌면을 판정했을 때, 해당 충돌면의 법선 벡터가 월드의 Up 벡터와 45도 이상 차이나지 않으면 플레이어는 지면 위에 있는 것으로 본다. 이럴 경우, 플레이어는 미끄러지지 않아야 한다.(아래로 이동하는 것을 무효화한다는 것) 플레이어에게 적용되는 중력 가속도는 다시 0으로 변경한다.  현재 프레임워크의 처리 순서는 입력 처리 -> 업데이트 -> 지연 업데이트 -> 렌더 순이다. 플레이어가 점프 키를 눌렀다고 말해주는 것은 입력 처리때 한다. 업데이트에선 플레이어가 총을 쏠 수 있는지, 체력은 0 보다 많은지, 애니메이션은 어떤 것을 재생해야 하는지 등을 정한다. 지연 업데이트에서 충돌검사와 충돌처리, 이동 및 회전을 한다. 이 때, 점프를 처리해야 할 것이다. 이후에 키 입력 트리거를 초기화한다.  지연 업데이트의 순서를 정확히 정해두자. 먼저 키 입력에 따른 회전 및 이동을 통해 진행방향을 정한다. 이후 주변 지형들과 충돌검사한다. 이 때, 충돌검사를 할 충돌체들을 구하기 위해 살짝 큰 Sphere 충돌체와 충돌하는 지형 충돌체들을 리스트로 받는다. 이들을 대상으로 위에서 말한 방식으로 진행방향 보정을 한 뒤, 다시 충돌하는지 검사한다. 이는 최대 처음 리스트의 개수만큼만 반복한다.(무한루프에 빠지는 것을 막기 위해) 이 충돌처리를 하는 과정에서 지면과 충돌했음을 알게 되면 IsGround를 True로 만든다. 지면과 충돌하지 않으면 False로 정한다. 지면 위로 올려주는 것도 해야 하는데. | | | |