**Retreat Protocol**

**플레이어 캐릭터: 원거리형 기술 문서**

본 문서에서는 플레이어가 게임 중 선택할 수 있는 세 가지 스타일 중 하나인 원거리형 스타일의 구현 과정에 대해 기술한다.

**무기**

원거리형 스타일은 다음과 같은 네 가지 무기를 사용할 수 있다:

|  |  |  |
| --- | --- | --- |
| **1** | **철갑소이탄** | 짧은 간격으로 두 번 발사하는 2연장 함포. |
| **2** | **근접유도탄** | 가까운 적을 추적하여 큰 피해를 입히는 2연장 함포. |
| **3** | **EMP자폭탄** | 주변의 적을 탐지하면 폭발하여 피해를 입히면서 밀어내는 포탄. |
| **4** | **플라즈마 포** | 2초 동안 에너지를 충전하여 강력한 피해를 입히는 레이저 무기. |

각 무기의 최대 사거리에 따로 제한이 있진 않지만, 포탄이 플레이어로부터 300거리 이상 떨어진 경우 게임에서 제거된다.

**무기1 – 철갑소이탄**

조준점을 기준으로 x축에서 ±1만큼 떨어진 위치에 서로 평행하게 날아가는 두 개의 포탄을 0.2초 간격으로 두 번 발사하여 총 4발의 포탄을 발사한다. 철갑소이탄에 명중한 적은 4초동안 방어력이 20% 감소한다.

철갑 소이탄에 적중한 적의 방어력을 떨어트린 후, 4초 후에 다시 방어력을 복구해야 하는데, 이 시점엔 이미 철갑소이탄이 적중하여, 방어력을 복구할 포탄 오브젝트가 사라진 상태이다. 이를 해결하기 위해 적의 코드에서 방어력을 초기화하는 함수를 구현하고, 해당 함수를 호출하는 방법으로 구현한다.

|  |
| --- |
| class 철갑소이탄\_무기  {      void 투사체\_발사      {          투사체\_생성 함수 호출          0.2초간 대기          투사체\_생성 함수 호출      }      void 투사체\_생성      {          for( int i = -1 ; i < 2 ; i += 2 )          {              카메라.position.x + i 위치에 카메라와 동일한 rotation을 가지는 투사체 생성          }      }  }  class 철갑소이탄\_포탄  {      void Update()      {          바라보는 방향으로 이동한다          플레이어와 자기 자신 사이의 거리를 계산          if( 현재 거리 > 300 ) 자기 자신을 삭제      }      void OnTriggerEnter(Collider 적)      {          적.HP -= 포탄 공격력          적.방어력 = -20          적.방어력\_복구(4) 함수 호출          자기 자신을 삭제      }  }  class 적  {      void 방어력\_복구(int 대기시간(초))      {          대기시간(초) 동안 대기          방어력 초기화      }  } |

**무기2 – 근접유도탄**

조준점을 기준으로 x축에서 ±1만큼 떨어진 위치에 서로 평행하게 날아가는 두 개의 포탄을 발사하여 총 2발의 포탄을 발사한다. 근접유도탄은 날아가는 도중 주변의 적을 발견하면 해당 적의 위치로 방향을 변경한다.

|  |
| --- |
| class 근접유도탄\_무기  {      void 투사체\_발사      {          for( int i = -1 ; i < 2 ; i += 2 )          {              카메라.position.x + i 위치에 카메라와 동일한 rotation을 가지는 투사체 생성          }      }  }  class 근접유도탄\_포탄  {      void Update()      {          바라보는 방향으로 이동          if( 적 발견 = false ) 적\_탐색 함수 호출      }      void 적\_탐색      {          Enemy 태그를 가진 모든 오브젝트를 검색하여 적\_리스트 생성          foreach( 적 in 적\_리스트 )          {              자기 자신과 오브젝트 사이의 거리를 측정(Vector3.Distance)              if( 거리 < 탐색 거리 ) LookAt(오브젝트)          }      }      void OnTriggerEnter(Collider 적)      {          적.HP -= 포탄 공격력          스스로를 삭제      }  } |

**무기3 - EMP자폭탄**

조준점 방향으로 정확하게 날아가는 하나의 포탄을 발사한다. EMP자폭탄은 날아가는 도중 주변 의 적을 발견하면 폭발하며, 폭발에 휘말린 적들을 피해를 입고, 폭발의 중심부로부터 외각 방향으로 밀쳐진다.

|  |
| --- |
| class EMP자폭탄\_무기  {      void 투사체\_발사      {          EMP자폭탄 오브젝트를 생성하여 바라보는 방향으로 rotation 설정      }  }  class EMP자폭탄\_포탄  {      void Update()      {          바라보는 방향으로 이동한다          적\_탐색 함수 호출          플레이어와 자기 자신 사이의 거리를 계산          if( 현재 거리 > 300 ) 스스로를 삭제      }      void 적\_탐색      {          Enemy 태그를 가진 모든 오브젝트를 검색하여 적\_리스트 생성          foreach( 적 in 적\_리스트 )          {              자기 자신과 오브젝트 사이의 거리를 측정(Vector3.Distance)              if( 거리 < 탐색 거리 ) EMP폭발(적\_리스트) 함수 호출          }      }      }      void EMP폭발(list 적\_리스트)      {          폭발 연출용 오브젝트를 생성          foreach( 적 in 적\_리스트 )          {              자기 자신과 오브젝트 사이의 거리를 측정(Vector3.Distance)              if( 거리 < 폭발 거리 )              {                  적.HP -= 포탄 공격력                  폭발의 외각 방향으로 밀쳐낸다(AddForece.Impulse)              }          }          스스로를 삭제      }  } |

**무기4 – 플라즈마 포**

발사 시 조준 방향을 기억하고, 2초 후 기억한 방향으로 발사되어 접촉하는 모든 적에게 피해를 입히는 레이저 포를 발사한다.

일반적인 포탄과 달리 투사체가 아니고, 접촉하는 모든 적에게 피해를 입혀야 하기 때문에 BoxCollider 컴포넌트를 가진 광선 오브젝트를 생성하는 방식으로 구현한다.

기본적으로 BoxCollider는 박스의 한 가운데를 중점으로 삼기 때문에 광선이 플레이어로부터 양 옆으로 발사되는 모양을 보이는데, 이는 Center 값을 조절함으로써 콜라이더의 중점 위치를 변경하여 해결할 수 있다.

|  |
| --- |
| class 플라즈마포\_무기  {      void 광선\_발사      {          현재 카메라의 각도 저장          2초간 대기          저장해둔 각도와 동일한 rotation을 가지는 광선 오브젝트 생성      }  }  class 플라즈마포\_광선  {      void Start()      {          4초간 대기 후 스스로를 삭제 (Invoke)      }      void OnTriggerEnter(Collider 적)      {          적.HP -= 광선 공격력          스스로를 삭제      }  } |

**스킬**

원거리형 스타일은 다음과 같은 두 가지 스킬을 사용할 수 있다:

|  |  |  |
| --- | --- | --- |
| **Q** | **전자기장** | 함선 주변에 있는 적 포탄의 이동을 멈춘다. |
| **E** | **비상 발전** | 다음 번에 사용하는 무기의 대기시간을 제거하여 곧바로 충전시킨다. |

**스킬1 – 전자기장**

사용 시 씬 내의 적 포탄을 모두 검색한 후, 플레이어 함선 주변의 포탄만 골라 이동속도를 0으로 만들어 이동을 멈추게 한다. 사용 후 12초가 지나면 스킬을 재사용할 수 있다. 전자기장에 의해 이동이 멈춘 포탄은 시간이 지나도 이동속도를 회복하지 않는다.

|  |
| --- |
| void 사용  {      EnemyBullet 태그를 가진 모든 오브젝트를 검색하여 적포탄\_리스트 생성      foreach( 적\_포탄 in 적포탄\_리스트 )      {          플레이어와 적\_포탄 사이의 거리를 측정 (Vector3.Distance)          if( 측정 거리 < 기준 거리 ) 적\_포탄.moveSpeed = 0      }  } |

**스킬2 – 비상 발전**

사용 시 플레이어는 비상 발전 상태에 돌입하고, 비상 발전 상태에서 발사하는 첫번째 무기의 대기시간을 즉시 0으로 만들어 제거한다. 이후 비상 발전 상태를 해제한다. 사용 후 (비상 발전 상태에 돌입한 순간으로부터) 12초가 지나면 재사용할 수 있다.

비상 발전을 사용한 후 12초 동안 아무런 무기도 발사하지 않고 있다가, 한 번 더 비상 발전을 사용하더라도 대기시간이 두 번 제거되진 않는다.

플레이어가 무기를 선택하여 발사한 후, 무기 코드에서 스스로 자신의 대기시간(쿨타임)을 초기화 하는 게 아니라, 플레이어 코드에서 방금 발사한 무기의 대기시간을 초기화하는 함수를 호출해주는 구조로 구현되어 있다. 여기서, 비상발전을 사용한 후에는 쿨타임을 초기화하지 않도록 조건문을 추가하는 방법으로 구현한다.

|  |
| --- |
| class 비상발전  {      void 사용      {          플레이어.비상발전\_버프 = true      }  }  class 플레이어  {      void 공격      {          선택된\_무기.포탄\_발사 함수 호출          if( 비상발전\_버프 == false ) 선택된\_무기.쿨타임\_시작 함수 호출          else 비상발전\_버프 = false      }  } |