**파티클시스템 프레임드롭 문제 해결방안**

**파티클시스템을 사용할 시, 프레임드롭이 발생하는 현상이 있다. 이를 해결하기 위해 파티클시스템의 부하를 줄여야 한다. 파티클시스템을 사용할 때에는 이하를 확인 할 것.**

1. emitter 수를 줄인다.

2. 투명도 요구사항을 줄인다(Color탭에서 alpha값, 렌더러의 머티리얼 alpha값).

3. 동적 배칭을 사용한다.

<https://docs.unity3d.com/kr/current/Manual/DrawCallBatching.html>

4. 오클루전컬링을 사용한다(Window->Occlusion Culling).

<https://docs.unity3d.com/kr/530/Manual/OcclusionCulling.html>

5. 최소 최대 곡선을 최적화한다.

6. minMaxCurve(Emission)는 constant형으로 한다(어쩔 수 없이 커브형일경우 노드를 줄여야한다).

7. 텍스쳐 아틀라스를 활용한다(아티스트가 없어 실사용은 불가능할 것으로 예상된다).

**프레임드롭이 발생할 때를 포함하여 최저프레임률로 프레임을 고정시킨다면 프레임드롭은 발생하지 않는다. 다만, 매우 낮은 프레임의 게임이 되지 않으려면 최저프레임을 높이는 작업이 필요하다. 따라서, 전체적인 최적화 작업이 중요하다. 최적화를 위해 프로젝트 진행 중 아래를 확인하여 실천 할 것.**

1. Find류의 함수는 사용을 최소화 한다.

2. 로딩씬을 활용하면 필요한 리소스를 게임 시작 전에 미리 로딩 할 수 있다.

3. 객체는 멤버 변수에 저장해서 캐싱하여 사용한다(커뮤니케이션 오버헤드를 줄임).

4. Instanitate 와 Destory 는 사용 최소화 한다(setActive 사용).

5. 박싱 언박싱을 최소화 한다.

6. 문자열은 가능하면 readonly 혹은 const 키워드를 사용한다.

7. 문자열 병합은 StringBuilder의 Append를 사용한다.

8. 되도록이면 foreach 보다는 for를 사용한다.

9. 태그 비교에서는 CompareTag()를 사용한다.

10. 되도록이면 비교문에서는 == 대신 .equals()를 사용한다.

11. 콜백 함수중 쓰지 않는 함수는 제거한다.

12. 텍스쳐 사이즈는 2의 제곱.(900 x 900 -> 1024 X 1024)으로 맞춰주어야 한다.

13. 메쉬 콜라이더는 절대 사용하지 않는다.

14. 되도록이면 tag보다는 layer를 사용한다.

15. 라이팅세팅에서 Precomputed Realtime GI, Based GI, Fog 등을 전부 끈다.

16. 프레임드롭이 일어나도 상관 없는 때에 System.GC.Collect() 로 GC를 발동시켜준다.

17. 디버그로그는 프레임테스트 및 최종빌드시에는 반드시 제거하도록 한다.

18. 배경음악은 Load in Background 기능을 활용하면 좋다.

19. 캔버스 내 UI들은 한 번에 변경이 이루어지는 것 들 끼리 같은 캔버스로 묶는편이 좋다.

20. 모든 글자를 임포트 하지 말고 Dynamic폰트 혹은 Packed폰트를 사용한다.

21. 리소스폴더는 사용을 최소화한다.

22. 성질이 동일한 오브젝트들은 하나의 메쉬와 재질을 사용하도록 통합한다.

23. 움직이지 않는 오브젝트들은 static으로 설정해서 배칭이 되게 한다.

24. 동적배칭을 사용하도록 체크한다.

25. 셰이더는 되도록 모바일용셰이더를 사용한다(가장 빠른 셰이더는 VertexLit이다).

26. 움직이는 충돌체는 반드시 리지드바디 컴포넌트를 넣는다(isKinematic사용).

27. Solver Iteration Count를 되도록 낮게 조정한다(Edit -> Project Setting -> Physics).

28. Rigidbody Sleep을 사용한다.

29. profiler를 활용하여 수행시간이 높은 부분을 집중적으로 수정한다.

※ CPU Usage 란에서, GC Alloc 기준으로 정렬하면, 누가 할당을 언제 시도하는지 볼 수 있습니다. 할당을 최대한 줄이고 몰아서 하면, 게임이 부드러워집니다. physics.engine 할당 같은 것은 제어할 수 없지만, 그 정도는 괜찮죠. Memory Area 에서는, 현재 메모리 상태의 스냅샷을 찍을 수 있습니다. 어디에서 메모리 누수가 발생하는지 파악가능합니다.

30. Application.targetFrameRate 를 사용하여 프레임을 고정시킨다.