코드리뷰 체크리스트

개요

Pull Request를 받고 코드리뷰할 때 확인해야하는 항목들을 나열한다.

basic checklist

빌드중

- (수정량이 많은 경우) 모든 플랫폼에 대해 컴파일/쿠킹 테스트가 되었는가?

파일

- 브랜치와 관계 없거나, 임시/중간 파일을 올리지는 않았는가?
- 처음 등록되는 확장자의 큰 용량의 파일이 git-lfs 관리 대상 (.gitattribute) 으로 등록 되었는가?

코드 정리

- Indent를 Tab으로 사용했는가?
- Indent의 Depth가 깨진 부분은 없는가?
- 외부에서 가져오거나(수식 포함), 이해하기 어려운 코드에 주석이 적절하게 달려있는가?

- 꼭 필요한 곳에만 주석을 달았는가? 주석이 너무 길지는 않은가?
- 임시 코드에 #todo-ovdr 해쉬태그를 포함한 주석이 달려있는가?
- 더이상 사용되지 않거나, 필요없는 공백/코드/주석이 있지는 않는가?
- 헤더파일의 변수/함수들이 기능/접근지정자/키워드(static 등) 에 따라 분리 되었는가?
- 한 줄의 코드에서 너무 많은 연산을 하고 있지는 않는가? 적절한 가독성으로 임시 변수에 담아 개행할 수는 없는가?
- 한 줄의 긴 코드에서 컴마, 논리 연산자 등에 따라 적절한 가독성으로 개행 되었는가?
- 한 함수에서 너무 많은 역할을 하고 있지는 않는가?
- Guard Clause를 적극 사용해 중첩 블록을 줄여 가독성을 높였는가?

C++ Common

- UE 코딩 표준을 잘 따랐는가?
- 변수명이 적절한가? 축약어를 사용하지는 않았는가?
- 목적/크기에 맞는 변수 타입/자료구조가 사용되었는가?
- auto를 사용하지는 않았는가? 이유가 있다면, 레퍼런스를 붙여 사용했는가?
- 람다 함수
  - 람다 함수를 적재하는 변수 타입에 auto를 사용했는가?
  - 람다 함수/함수포인터에 변수를 넘길 때, 해당 변수가 사라질 가능성은 없는가? 사라질 가능성이 있는 변수에 Weak Pointer를 사용했는가?
  - 람다 함수 사용시 스코프의 모든 변수를 무분별하게 캡쳐하지는 않았는가?
- 캡슐화된 함수/변수를 임의로 풀어서 외부에서 사용하지는 않았는가?
- 다형성 의도가 있는 최상위 클래스의 소멸자에 virtual을 붙였는가?
- 기존 함수에 virtual 키워드가 붙었다면, 그 이유는 합당한가?
- 생성자/소멸자에서 virtual 함수를 부르지는 않았는가?
- 변수를 사용하기 전에 nullptr, IsValid(), IsEmpty() 등 sanity check 되었는가?
- for, while 루프 안에서 자료구조의 원소를 삭제하지는 않았는가?
- 자료구조의 원소를 Raw Pointer/레퍼런스로 들고 있지는 않는가?
- #pragma optimize("", off) 를 사용하지는 않았는가?
- 재귀 호출을 사용하지는 않았는가? (stack overflow)
- 헤더에서 헤더를 include 하지는 않았는가? 전방선언으로 대체할 수는 없는가?
- 헤더 파일에서 변수 초기화를 하지는 않았는가?
- const 멤버함수 및 파라메터를 적극적으로 사용했는가?
- 상수를 그대로 쓰지 않고 constexpr을 적극적으로 사용했는가?
- 8 byte가 넘는 매개변수 타입에는 const &를 사용했는가?

- c++ 캐스팅을 적절하게 사용했는가? 다운캐스팅은 없는가?
- 나눗셈에서 0으로 나누는 경우는 없는가?

UE5 Common

- 언리얼 엔진을 위한 에픽 C++ 코딩 표준을 잘 따랐는가?
- UCLASS에 생성자/소멸자에서 게임 로직을 작성하지는 않았는가?
- UCLASS에 const FObjectInitializer& 를 파라메터로 받는 생성자를 만들었는가?
- 언리얼 애셋 경로를 하드코딩해서 레퍼런스 하지는 않았는가? DefaultGame.ini에 등록해서 쓸 수 없는가?
- GameThread가 아닌곳에서 UObject를 사용하지는 않았는가?
- UObject 파생 타입에 TObjectPtr를 썼는가? Raw pointer를 쓰지는 않았는가?
- Raw pointer 대신 언리얼 Smart Pointer를 적극적으로 사용했는가?
- GC 대상인 멤버 변수에 UPROPERTY 또는 TWeakObjectPtr가 붙었는가? 특히 USTRUCT 및 멤버변수가 GC 대상인가?
- UPROPERTY가 붙는다면, 해당 클래스에서 생명 주기를 관리하는 대상인가? TWeakObjectPtr로 대체할 수는 없는가?
- 런타임용 UPROPERTY 에는 `Transient` 사용. 불필요한 저장/복제 (`SaveGame` / `Replicate`) 최소화
- #define LOCTEXT_NAMESPACE 이후 #undef 했는가?
- FSimpleDelegateGraphTask::CreateAndDispatchWhenReady를 사용하지는 않았는가?
- UE_LOG 사용 시 LogTemp가 아닌 적절한 로그 범주를 사용했는가? (Output log 창에 스팸되지 않도록 자주 볼 필요가 없는 내용은 VeryVerbose 사용)
- EComponentMobility 가 목적에 맞게 설정되어 있는가?
- NativeTick 이 필요없는 Widget에는 UCLASS(meta=(DisableNativeTick)) 이 붙어있는가?
- gameEvent delegate 사용 시, 호출 순서가 보장 되는가?

Dev vs Shipping

- check() 구문 안에 쉬핑 빌드에서도 실행 되어야하는 로직을 짜지는 않았는가?
- if, for 문 등에 중괄호가 없는 상태로, check / UE_LOG 등 쉬핑에서 사라지는 매크로를 사용하지는 않았는가?

UE5 Slate UMG

- Slate Update 최적화 가이드 라인 및 하위 문서를 잘 따랐는가?
- Slate Batching 최적화 가이드 라인 및 하위 문서를 잘 따랐는가?

Multi-Thread Common

- 멀티스레드가 꼭 필요한 상황인가? 안전하게 구현하기 위한 노력과 유지보수 비용을 감수할 가치가 있는가?
  - 참고) 큰 컨텍스트 스위칭 비용과 동기화를 위한 오버헤드로 인해 상용 미들웨어/엔진의 구현에서도 최적화 효율이 70% 이상 나오기 힘듦
  - 모바일에서는 발열로 인한 스로틀링과 배터리 소모로 인해 최적화 효율이 더 크게 떨어질 것으로 추측됨
- Race condition이 최대한 발생하지 않도록 공유되는 메모리를 최소한으로 사용했는가?
- 이벤트 기반으로 동기화를 하고 있는가? sleep으로 동기화를 하고 있지는 않는가?
- sleep(0)을 매 틱 호출하고 있지는 않는가? (모바일에 치명적)
- spin lock을 사용하지는 않는가? (모바일에 치명적)
- 스레드가 충분히 연속된 일을 하는가? 잠깐 일하고 쉬기를 반복하지는 않는가? (모바일에 치명적)
- OS 스레드를 꼭 생성해서 사용해야 하는가? 언리얼 스레드 태스크로 대체할 수 있지는 않는가?
- 스레드는 풀링 되고 있는가? 또는 작업마다 스레드를 생성하고 파괴하지는 않는가?

CPU Performance

- Tick
  - 비어있는 Tick이 계속해서 호출되고 있지는 않는가?
  - 꼭 필요할 때만 이벤트를 받아서 Tick을 활성화/비활성화 해주고 있는가?
  - 매 Tick 마다 할 필요 없는 연산을 매번 하고 있지는 않는가?
  - Tick 함수 로직 중 이벤트 기반으로 만들 수 있는 것은 없는가?
  - 인터벌이 필요한 곳에 FTickFunction::TickInterval 로 틱 호출 간격을 주었는가?
  - 다중 루프를 구현할 때, 소량 루프가 대량 루프를 포함하도록 구성했는가?
  - SkeletalMesh가 사용된 것 중 StaticMesh로 대체할 수 있는 것은 없는가?
- Hitch
  - 로딩 타임이 아닌 런타임에서 LoadObject, StaticLoadObject, LoadSynchronous를 사용하지는 않았는가?
  - 런타임에 사용되는 애셋 UPROPERTY 타입에 하드 레퍼런싱을 사용하지는 않았는가?
  - UActorComponent::bCanEverAffectNavigation 가 true로 설정되어있지는 않은가?
- Math
  - 나눗셈 연산을 곱셈 연산으로 대체할 수는 없는가?
  - Distance를 DistSquared로 대체할 수는 없는가? (Distance, Normalize, Length 반복 사용으로 인한 성능 비용 최소화)
- Container
  - TInlineAllocator를 사용할 수는 없는가?

    - Reserve로 메모리를 미리 할당해 두었는가?
    - Add 대신 Emplace를 사용했는가?
- FString
    - 함수 인자로 받을 때 등, FString에 대한 깊은 복사가 항상 일어나지는 않는가? FString 대신 Raw pointer로 받아 쓸 수 없는가?
    - 비교/키로 쓰는 문자열은 `FName` 으로 사용 (단순 index 비교), 빈번한 `FString` → `FName` 변환 반복 금지
- FName
    - FText::FromStringTable(), LOCALIZATION_StringTable 및 FString → FText 의 묵시적 변환을 사용하지는 않는가? FText 서치 부하 줄이기.
- Pointer
    - TWeakObjectPtr과 Raw pointer를 비교할 때, .Get으로 Raw pointer를 얻어와서 비교했는가?
- Physics
    - 시뮬레이션 대신 Tween, Raycast, Overlap, PredictProjectile 등 Async 쿼리로 처리할 수는 없는가?
    - 런타임에 히칭을 유발할 수 있는 Rigid Body를 동적 생성 or Collision 설정 변경을 하지는 않았는가?

Memory Andrew Kelley Practical Data Oriented Design (DoD)

- 동적 할당을 최대한 줄이고 스택 활용
    - 힙 메모리 할당은 OS 커널을 거쳐야 하고 단편화로 인한 캐시 미스로 성능 문제를 크게 유발합니다
- 키 기반 O(1) 조회가 꼭 필요한 경우가 아니라면 `TMap` 대신 `TArray` 사용
    - 랜덤 메모리 접근보다 순차 접근 위주로 사용시 캐시 히트에 유리, 크기를 미리 정해두면 동적 할당 최소화 가능
    - 수십, 수백개 정도의 작은 데이터에서 캐시라인 이내의 순차 검색이나 사전 정렬된 Array는 캐시 히트로 인해 키 기반 검색보다 빠를 수 있음
    - 원소가 크고 랜덤 키 조회 위주에 데이터가 수백개 이상이면 `TMap` 이 점점 유리해짐
- 객체 풀링 (오브젝트/컴포넌트/버퍼)으로 런타임에 동적 Spawn/Destroy 반복 최소화
- 구조체 반복 복사 , 재할당 최소화 , 8바이트 초과 파라미터는 `const&` 로 참조 전달
- 순환 참조(UObject ↔ UObject, UMG 위젯 등)가 없는가? 필요 시 한쪽을 `TWeakObjectPtr` 로 약화했는가?

- 메모리 침범(Overrun) 방지를 위해 배열/포인터 접근 시 인덱스 범위 체크 (`check()` , `IsValidIndex()` 등).
- `memreport -full` / `obj list` 로 기능 추가 전,후의 메모리 스냅샷 비교. 메모리 누수 or 좀비 오브젝트 색출 (GC 후 비교해야 정확함)
- 멤버 변수 선언 위치에서 메모리 Alignment 를 고려했는가?

Network Performance

- 꼭 필요한 Component만 Replication 하고 있나? 로컬 또는 서버에서만 생성할 수는 없는가?
- 꼭 필요한 변수만 Replication 하고 있나? 로컬 또는 서버에서만 처리할 수는 없는가?
- 매 Tick 변경되는 값이 Replication 되고 있지는 않은가?
- Tick Interval 을 두고 천천히 동기화 해도 되는 변수인가?
- 매 Tick RPC가 호출되고 있지는 않은가?
- Net Cull Distance가 적절히 설정 되었나?
- 변수의 DOREPLIFETIME이 적절히 설정 되었나? 최초 한번만 동기화 되는 변수이면 Initial Only 로
- UPROPERTY의 Reliable attribute가 꼭 필요한 변수에만 사용되었는가?

GPU Performance

- 셰이더에서 UV Input에 CutomizedUV를 사용했는가?
  - Custom UV 로 입력이 변조되면서 랜덤 접근시 캐시 히트가 어려워지면서 큰 성능 비용이 발생함
- A채널을 꼭 써야 하는가? (A채널 하나가 RGB 채널을 다 합한 만큼의 데이터를 사용함)
- OMR 패킹에 (AO/Masked/Roughness/Specular 등 채널 합치기) 불필요하게 빈 채널이 있는가? 있다면, Greyscale 맵 여러개로 분리
- Use Full Precision 이 체크되어 있지는 않은가?
- 동적 분기가 사용되지는 않았는가?
- CPU에서 한 번의 연산으로 할 수 있는 것을 GPU에서 모든 픽셀에 대해 하고 있지는 않은가?
- 매 틱 GPU에 업데이트 하는 변수가 있지는 않은가?
- 각 노드 아웃풋이 필수가 아닌데(특히 Roughness) 사용되고 있지는 않은가?

Unreal Engine

- unreal-engine 수정 가이드 를 잘 따랐는가?

logical checklist

WIP

Bolier Plate

- 중복되는 코드는 없는가? 공통 로직은 함수/인터페이스 등으로 재사용 할 수는 없는가?
- 함수 인자가 너무 길지는 않은가? 구조체 등으로 줄일 수는 없는가?

structurical checklist

WIP