TL; DR

TODO:

0.1 Structure

- `EffectableComponent`**s** are `ActorComponent`s that allow for delegation (effects). They have predefined places that allow for code modification.
  - Let's use `StatsComponent` as an example. Say we want a Pokémon-style "Adamant" nature (+10% PhA/−10%SpA). One such place for modification is in the function `RecalculateStats`.

```
void UStatsComponent::RecalculateStats(const bool bResetCurrent)
{
    for(FStat* Stat : StatsArray)
    {
        ExecuteBeforeRecalculateStats(Stat, bResetCurrent);
        Stat->Update(GetLevel(), bResetCurrent);
        ExecuteAfterRecalculateStats(Stat, bResetCurrent);
    }
}
```

- **Delegate arrays** are variables inside of `EffectableComponent`s. They hold functions that execute when needed.
  - Let's use `StatsComponent`'s `AfterRecalculateStatsArray` in our example. In this case, after stats are recalculated (say, on level-up), the base PhA would increase by 10% and the base SpA would decrease by 10% (additively):

```cpp
// Define "adamant" delegate (+10% PhA/-10% SpA)
UStatsComponent::FRecalculateStatsDelegate AdamantRecalculateDelegate;
AdamantRecalculateDelegate.BindLambda( InFunctor: [StatsComponent](FStat* Stat, bool bResetCurrent) ->void
{

    // +10% PhA
    if ( Stat->Name() == StatsComponent->PhysicalAttack.Name())
    {
        Stat->ModifyValue( Modifier: 10, EStatValueType::Permanent, EModificationMode::AddPercentage);
        if (bResetCurrent)
            Stat->ModifyValue( Modifier: 10, EStatValueType::Current, EModificationMode::AddPercentage);
    }

    // -10% SpA
    if ( Stat->Name() == StatsComponent->SpecialAttack.Name())
    {
        Stat->ModifyValue( Modifier: -10, EStatValueType::Permanent, EModificationMode::AddPercentage);
        if (bResetCurrent)
            Stat->ModifyValue( Modifier: -10, EStatValueType::Current, EModificationMode::AddPercentage);
    }

});
StatsComponent->AfterRecalculateStatsArray.Add(AdamantRecalculateDelegate);
```

Table 1: Type Trait Suggestions

A	B