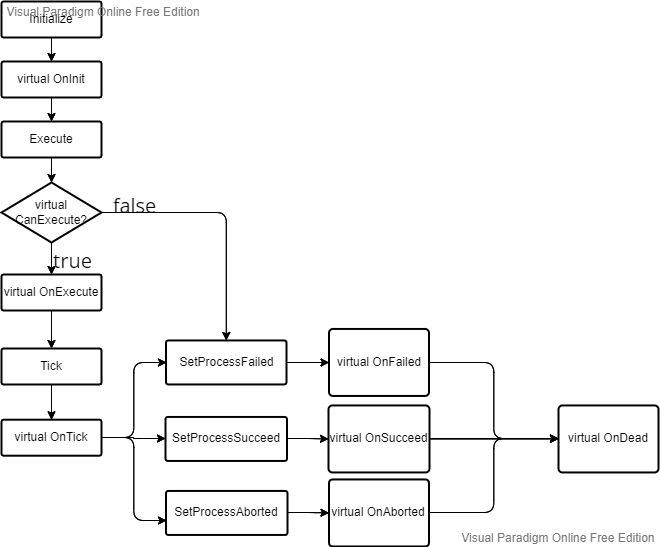
UE4 ARPG GAME Demo 정리문서

액션RPG 게임내 주요기능들의 구현방식을 요약하여 설명 드리는 문서입니다。

프로젝트 주소： [*https://github.com/kklmg/traveler-ue4-Demo-*](https://github.com/kklmg/traveler-ue4-Demo-)

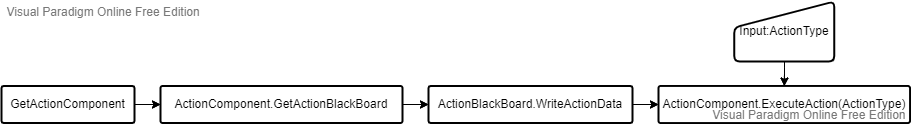
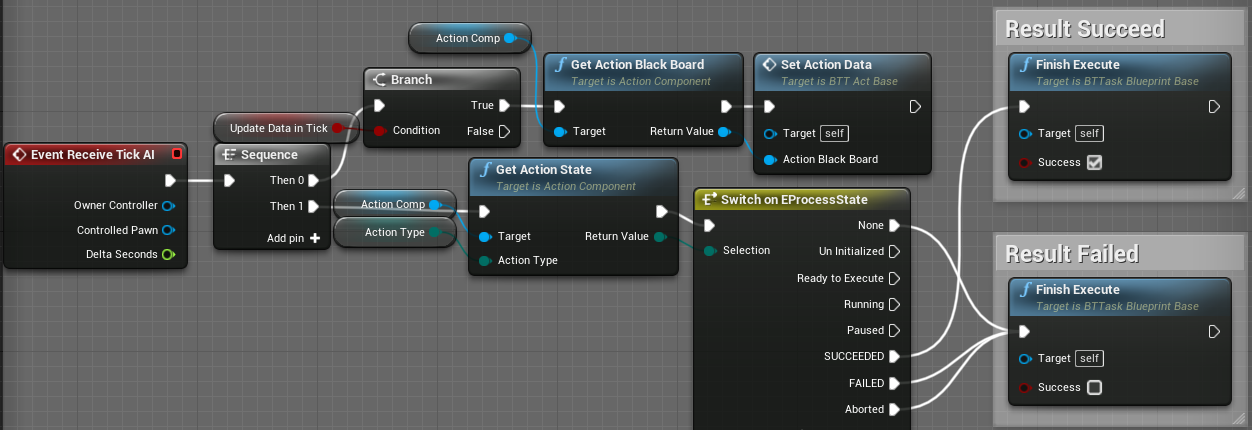
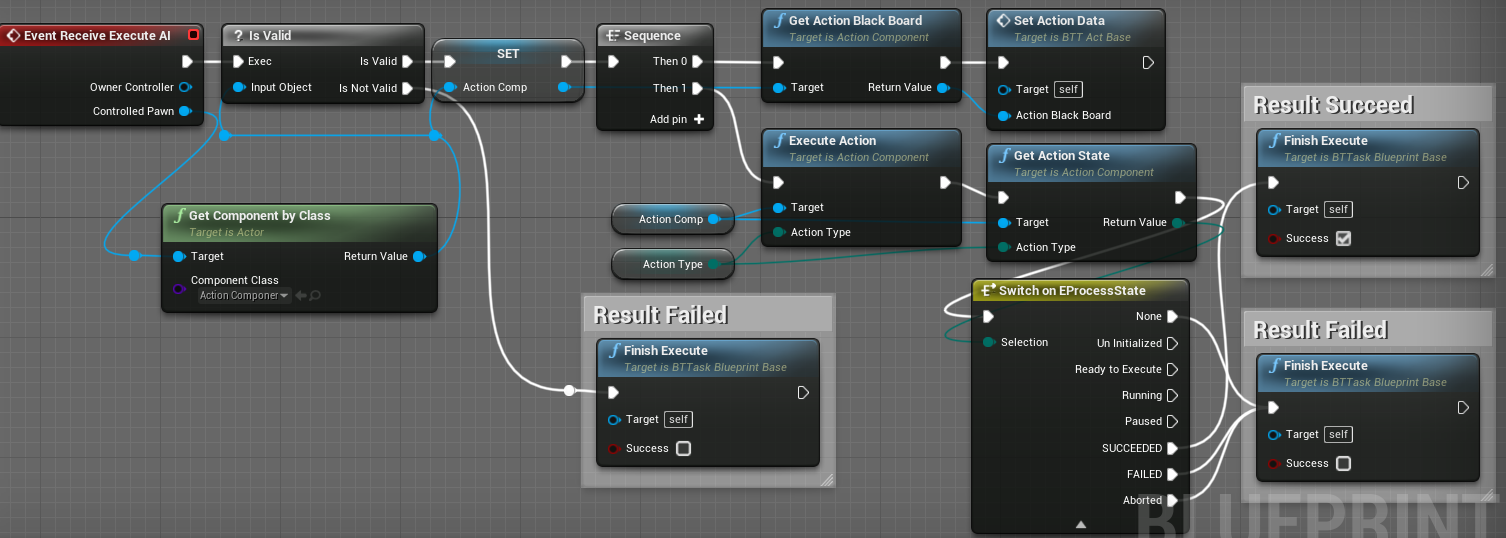
1. 프로세스(Process)

* 요약：   
  게임에서 캐릭터들의 액션들이 순식간에 완성되는것은 매우 드문일입니다。 이런 연속적으로 실행되는 코드구조가 필요하여 프로세스 클래스를 만들었습니다。
* 구현：

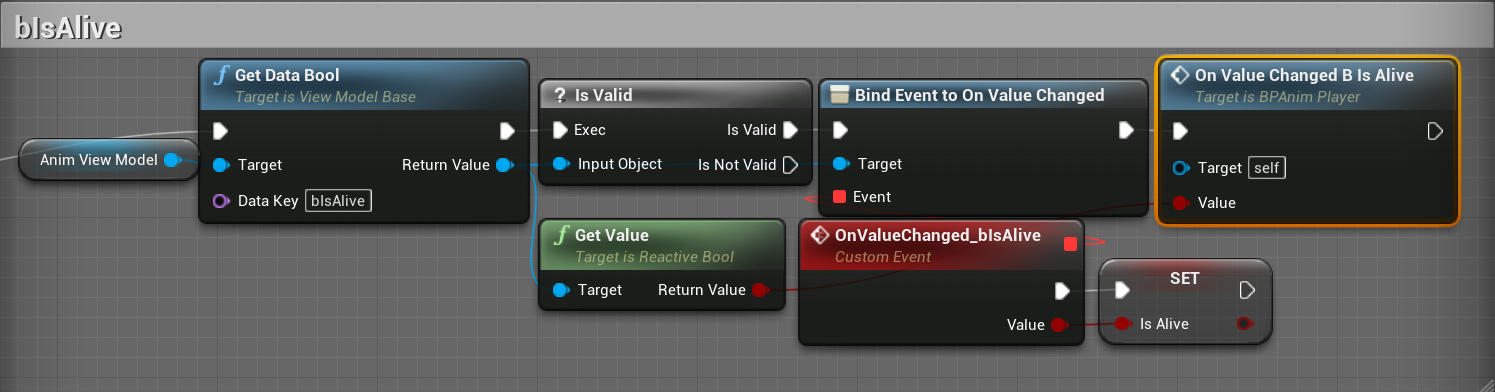
1. 프로세스 상태(EProcessState) :  
   프로세스의 상태를 표현하는 열거형입니다。 언리얼에서 제공하는 Behavior Tree 와 호환을 갖기 위하여 똑같이 Succeed,Failed,Aborted등 상태를 도입하였습니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Enums/EnumProcessState.h>
2. 프로세스 클래스(UProcessBase) :  
   모든 프로세스의 기본 클래스입니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Process/ProcessBase.h>
3. 프로세스 실행 흐름도：

# 2.액션시스템

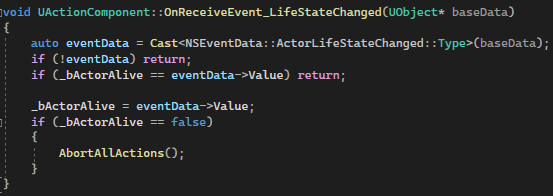
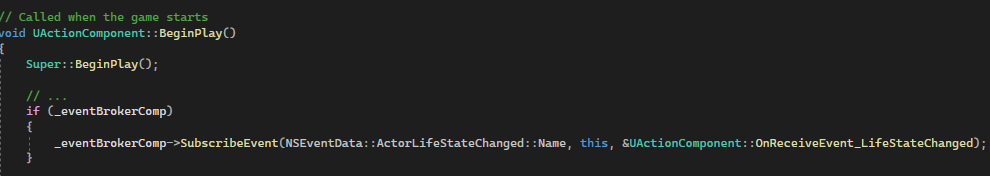
* 요약：  
  게임내 캐릭터들의 액션들을 에디터에서 자유롭게 조합할수 있고 액션들을 쉽게 컨트롤 하기 위하여 만들어진 시스템입니다。
* 구현：

1. UActionBase:  
   프로세스(UProcessBase)를 상속하여 캐릭터의 액션들을 구현하기 위하여 만든 특수한 프로세스입니다。 모든 캐릭터 액션의 기본 클래스입니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Actions/ActionBase.h>
2. UActionBlackBoard :  
   블랙보드패턴을 사용하여 액션을 실행하기 위한 임시 데이터를 저장하는 공간입니다。   
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Actions/ActionData/ActionBlackBoard.h>
3. UActionPreset:   
   에디터에서 세팅할수 있는 액션클래스들의 집합을 보유하고 액션 인스턴스를 생성，컨트롤하는 역할을 합니다。 액션타입(EActionType)을 입력받아서 해당되는 액션의 실행，중단，상태조회등 기능을 제공합니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Actions/ActionPreset/ActionPreset.h>
4. UActionPresetGroup:   
   보통 게임내 캐릭터가 다른 무브먼트 상태에서 할수있는 액션은 다를것입니다。 예를 들면 캐릭터가 지면상에서 있을때 점핑을 할수 있으나 수영하고 있을때는 점핑을 할수 없습니다。그러므로 캐릭터가 다른 무브먼트생태에서 서로 다른 행동을 할수있도록 ActionPresetGroup 클래스를 만들었습니다。  
   무브먼트모드(EMovementMode)와 액션션프리셋(UActionPreset)을 1대1로 매핑되게 저장하여 관리하도록 하고 액션타입(EActionType)을 입력받아서 현재 무브먼트상태와 매핑된 액션의 실행，중단，상태조회등 기능을 제공합니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Actions/ActionPreset/ActionPresetGroup.h>
5. UActionComponent:  
   액터의 액션들을 컨트롤할수 있는 최상위 관리자입니다.UActionPresetGroup과 UActionBlackBoard를 보유하고 컨트롤하는 기능을 제공합니다。 캐릭터의 무브먼트상태를 UActionPresetGroup에 알려줍니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/ActionComponent.h>
6. 일반적인 액션의 실행흐름도：  
   
7. 비헤이버트리에서 액션시스템을 사용하기（비헤이버 타스크 블루프린트）:  
     
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Content/MyAssets/AI/BTT_Act_Base.uasset>
8. UActionBase 를 상속하여 다양한 액션 제작：  
   <https://github.com/kklmg/traveler-ue4-Demo-/tree/master/Source/traveler/Public/Actions><https://github.com/kklmg/traveler-ue4-Demo-/tree/master/Source/traveler/Public/Weapon/WeaponAction>

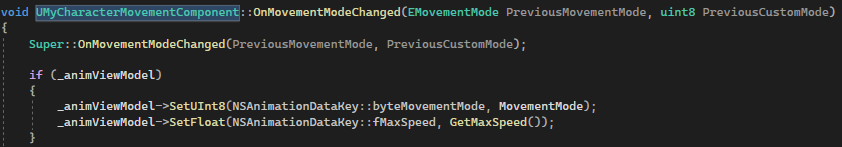
# 3. 뷰모델(View Model)

* 요약：   
  게임 실행중에 데이터는 계속 변경되므로 그와 연관된 뷰를 갱신할 필요가 있습니다。 데이터를 갱신하는 로직을 Tick함수에 집어넣으면 뷰가 많아짐에 따라 퍼포먼스 이슈가 생길수 있습니다。 그래서 데이터의 변화를 감지하여 자동으로 뷰에게 알려주는 구조가 필요하여 뷰모델 클래스를 만들었습니다。
* 구현：
  1. ViewModelBase:  
     새로운 뷰를 추가하였을 때 또 다시 그에 해당하는 뷰 모델을 만드는 것은 번거롭습니다。 그러므로 블랙보드패턴을 사용하여 데이터를 데이터 이름과 데이터를 매핑하여 저장하는 구조로 구현하여 다시 새로운 뷰 모델을 만들 필요가 없게 하였습니다。 기본적으로 데이터의 조회와 세팅 기능을 제공하고 데이터를 세팅하면 자동으로 바인딩된 데이터를 업데이트 해줍니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Data/ViewModelBase.h>
  2. 사용 예제:  
     캐릭터애니메이션 블루프린트에서의 사용예제입니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Content/MyAssets/Character/Player/Animation/BPAnim_Player.uasset>

# 4. 이벤트 브로커（Event Broker）

* 요약：   
  프로젝트가 커지면서 컴퍼넌트간의 결합도가 높아지게 되여 코드구조가 복잡하지기 시작하였습니다。 그러므로 Publish-Subscriber 패턴을 사용하여 이벤트 브로커 클래스를 만들어 컴퍼넌트간의 결합도를 줄였습니다。
* 구현：
  1. UEventBroker  
     이벤트 등록，이벤트 구독，이벤트 퍼블리싱 등 기능을 제공합니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Event/EventBroker.h>
  2. UEventBrokerComponent  
     UEventBroker를 wrapping하여 컴포넌트간의 통신을 담당하는 역할을 합니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/EventBrokerComponent.h>
  3. 사용예제  
     UActionComponent에서 사용된 예제입니다。  
     캐릭터의 생명상태 이벤트를 구독하여 캐릭터가 사망했을때 진행중인 액션을 중당시키는 코드입니다。  
       
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/ActionComponent.h>

# 5. 애니메이션

* 요약：   
  언리얼엔진에서 제공하는 애니메이션 블르프린트，블랜드스페이스，몽테이지，노티파이어 등을 배우면서 캐릭터 애니메이션을 제작하였습니다。
* 구현：
  1. UAnimControlComponent:  
     애니메이션을 쉽게 컨트롤 할수 있도록 만든 액터 컴퍼넌트입니다。 애니메이션 뷰모델을 통하여 애니메이션 블루프린터를 업데이트해줍니다。 애니메이션 몽테이지，애니메이션 노티파이어 등 을 관리하는 기능을 제공합니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/AnimControlComponent.h>사용예제：  
       
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/MyCharacterMovementComponent.h>
  2. 애니메이션 블랜드 스페이스，애니메이션 블루프린트를 이용하여 캐릭터 애니메이션（애니메이션 블루프린트）을 제작：  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Content/MyAssets/Character/Player/BPAnim_Player.uasset>
  3. UIKComponent:  
     라인 트레이싱을 사용하여 캐릭터의 발높이와 지면과의 높이 차이，지면의 노멀값을 애니메이션 블루프린트에 전달하여 캐릭터의 발높이를 조절하게 합니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/IKComponent.h>
  4. UAnimNotifyStateWithKey:  
     언리얼에서 제공하는 UAnimNotifyState를 상속하여 기존 기능을 보류하면서 UAnimControlComponent 에서 등록된 listener들에게 알려주는 기능을 추가하였습니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/AnimNotify/AnimNotifyStateWithKey.h>

# 6. 머티리얼,이펙트

* 요약：   
  언리얼엔진에서 제공되는 Material,Material Function,Material Layer,Niagara System 을 배우면서 게임내 이펙트를 제작하였습니다。
* 구현：
  1. UEffectPlayerBase:  
     UMaterialInstanceDynamic를 세팅하고 내부에서 원하는 대로 머티리얼 파라미터를 꾸미며  
     외부에게 PLayEffect,StopEffect 기능을 제공합니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Effet/EffectPlayerBase.h>
  2. UEffectControllerComponent:  
     UEffectPlayerBase와 EEffectType(Enum)를 매핑하여 보유하고 EEffectType(Enum)를 입력 받아 해당된 이펙트를 컨트롤하는 기능을 제공합니다. 그리고 액터의 MaterialInstance를 가져와서 UMaterialInstanceDynamic를 생성， 세팅하는 역할을 해줍니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/EffectControllerComponent.h>
  3. Dissolve 머티리얼 제작  
     캐릭터가 사망할때 사용하고 있습니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/tree/master/Content/MyAssets/Material/Dissolve/Basic>
  4. 언리얼에서 제공되는 머티리얼 레이어를 사용하여 캐릭터 머티리얼 제작：  
     캐릭터 기본 머티리얼, Dissolve 머티리얼레이어,BlendColor 머티리얼레이러를 블랜딩하여 제작하였습니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Content/MyAssets/Material/Character/MI_Player.uasset>
  5. 언리얼에서 제공하는 니아가라（NIagara） 시스템을 이용하여 일반화살，화염화살，물화살，화염，분수， 화염방사기， 화염볼 폭발 등 이펙트를 제작하였습니다  
     <https://github.com/kklmg/traveler-ue4-Demo-/tree/master/Content/MyAssets/Effect>
  6. 언리얼에서 제공되는 데칼컴퍼넌트를 사용하여 화살 임펙트 효과를 제작하였습니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/tree/master/Content/MyAssets/Material/Decal>

# 7.무기 시스템

* 요약：   
  캐릭터가 다양한 무기를 사용하여 전투할수 있도록 구현하였습니다。
* 구현：
  1. AWeaponBase:  
     액터 클래스를 상속하여 구현된 기본 무기 클래스입니다。무기를 장착，해제，커트롤하는 기능을 제공합니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Weapon/WeaponBase.h>
  2. ABowBase:  
     AWeaponBase 를 상속하여 활의 애니메이션과 컨트롤하는 기능을 제공합니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Weapon/BowBase.h>
  3. UWeaponComponent:  
     캐릭터의 무기장착，해제，무기정보조회 등 기능을 제공하는 액터 컴퍼넌트입니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/WeaponComponent.h>

# 8. 캐릭터 상태

* 요약：   
  캐릭터의 상태，속성등을 쉽게 컨트롤할수 있는 구조로 구현하였습니다。
* 구현：
  1. FStatusTableRow:  
     언리얼에서 제공하는 FTableRowBase 를 상속하여 캐릭터의 상태값을 테이블로 편집할수 있도록 만든 TableRow입니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Status/StatusTableRow.h>
  2. UStatusBase :  
     UObject 를 상속하여 구현된 기본 상태 클래스입니다。 캐릭터의 상태값을 테이블（FStatusTableRow）에서 가져오고 조회하는 기능을 제공합니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Status/StatusBase.h>
  3. UBasicStatus:  
     UStatusBase 를 상속하여 기존 기능을 보유하면서 캐릭터의 생명，에너지처럼 소모가 가능 한 특수한 상태클래스 입니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Status/BasicStatus.h>
  4. UStatusComponent:  
     캐릭터의 속성들을 세팅，조회，변경 등 기능을 제공하는 클래스입니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/StatusComponent.h>

# 9. 전투 시스템

* 요약：   
   게임중에 발생하는 데미지，상태이상，버프，디버프를 구현하기 위한 구조로 구현하였습니다。
* 구현：
  1. UStatusEffectData:  
     상태이상을 구현하기 위한 데이터입니다。   
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Data/StatusEffectData.h>
  2. UStatusEffectProcessBase:  
     UProcessBase를 상속하여 구현된 기본 상태이상 프로세스 클래스입니다。 UStatusEffectData 를 입력 받아 프로세스를 초기화하고 지속적으로 캐릭터에게 상태이상 영향을 줍니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Damage/StatusEffectProcessBase.h>
  3. UStatusEffectProcessManager:  
     캐릭터의 상태이상프로세스를 매니저하는 클래스입니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Damage/StatusEffectProcessManager.h>
  4. FDamageData:  
     기본 데미지 데이터 입니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Damage/DamageData.h>
  5. UDamageHandlerComponent:  
     FDamageData, UStatusEffectData를 입력 받아 캐릭터에게 주는 데미지와 상태이상 효과를 처리해주는 액터 컴포넌트입니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/DamageHandlerComponent.h>

# 10. 오브젝트 풀(Object Pool)

* 요약：   
  게임내 빈번하게 생성，소멸되는 오브젝트는 오브젝트 풀을 사용하여 관리하는것이 적당합니다。 오브젝트 풀을 사용하면 오브젝트의 인스턴스를 생성할때의 비용을 줄일수 있고 가비지 컬렉터의 부하를 줄일수 있습니다。
* 구현：
  1. IPoolableInterface:  
     오브젝트풀의 기능을 사용하기 위하여 상속해야할 인터페이스입니다。  
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Interface/PoolableInterface.h>
  2. UObjectPoolBase:  
     TScriptInterface<IPoolableInterface> 의 배열로 오브젝트를 보유하고 빈 슬롯의 인덱스를 배열로 기록하여 퍼포먼스를 향상하였습니다.   
     <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/GameSystem/ObjectPoolBase.h>

# 11. 프로젝타일(Projectile)

* 요약：  
  게임내 프로젝타일들을 쉽게 다루기 위하여 그와 연관된 클래스를 만들었습니다。
* 구현：

1. AProjectileActorBase:  
   게임내 모든 프로젝타일의 기본이 되는 클래스입니다。 언리얼에서 제공하는 UProjectileMovementComponent를 사용하고 오브젝트 풀에서 관리할수 있도록 IPoolableInterface를 상속하였습니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Actors/ProjectileActorBase.h>
2. AArrowActorBase:  
   AProjectileActorBase 를 상속하여 캐릭터의 화살에 필요 되는 기능을 추가로 구현하였습니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Actors/ArrowActorBase.h>
3. UQuiverComponent:  
   AArrowActorBase의 생성을 관리하는 매니저입니다。 내부에서는 오브젝트 풀을 사용하여 프로젝타일을 관리하고 있습니다。 ABowBase에 부착하여 사용하고 있습니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/QuiverComponent.h>
4. UProjectileThrowerComponent:  
   일정 시간간격에 맞추어 지속적으로 프로젝타일을 생성해주는 액터 컴퍼넌트입니다。 내부에서는 오브젝트 풀을 사용하여 프로젝타일을 관리하고 있습니다。 게임내 화염방사기에 사용되고 있습니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Components/ProjectileThrowerComponent.h>

# 12. 기타 주요 구현한 것들

1. UDamageDisplayer:  
   게임중 발생하는 데미지를 보여주는 역할을 합니다。 FDamageDisplayData 를 입력받아  
   Damage Widget 생성하여 화면에 보여줍니다。 Damage Widget는 내부에서 오브젝트풀로 관리하고 있습니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Damage/DamageDisplayer.h>
2. 언리얼 비헤이버트리 시스템을 사용하여 드래곤의 AI를 구현하였습니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Content/MyAssets/AI/BT_Dragon.uasset>
3. 언리얼 widget를 사용하여 다양한 UI를 제작하였습니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/tree/master/Content/MyAssets/UI>
4. UInputHandlerComponent, UInputPresetBase:  
   에에디터에서 쉽게 유저의 입력을 편집할수 있도록 만든 클래스입니다。  
   <https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Input/InputHandlerComponent.h><https://github.com/kklmg/traveler-ue4-Demo-/blob/master/Source/traveler/Public/Input/InputPresetBase.h>