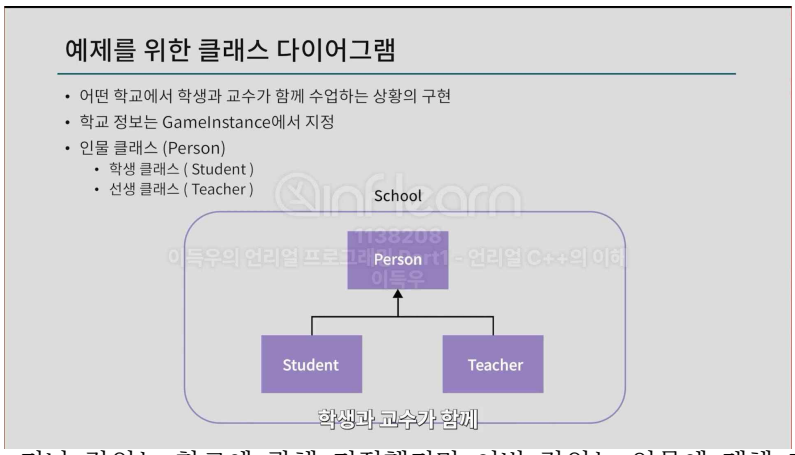
언리얼 프로그래밍 Part1-6

제목:언리얼 오브젝트 리플렉션 시스템 II

**강의 내용 : 언리얼 오브젝트 리플렉션 시스템의 활용

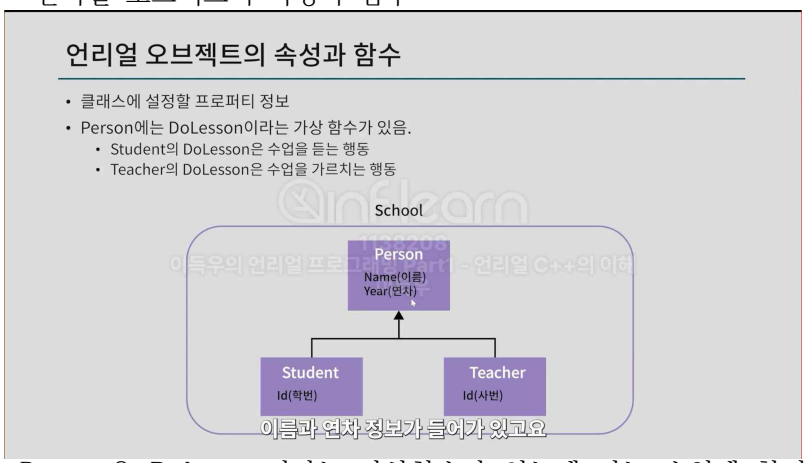
**강의 목표 : 언리얼 오브젝트 리플렉션 시스템을 사용해 언리얼 오브젝트를 다루는 방법의 학습

**실습 예제의 구성



-지난 강의는 학교에 관해 지정했지만 이번 강의는 인물에 대해 지정해줄거임.
-학생과 선생은 인물을 상속받는 구조로 만들어 볼 것임

**언리얼 오브젝트의 속성과 함수



-Person은 Dolesson이라는 가상함수가 있는데 이는 수업에 참여하는 행동을 지정하는데 사용될 가상함수이다.
-Student는Dolesson에서 수업을 듣는행동 Teacher는 수업을 가르치는 행동을 구현할 것임.

**예제 코드

-MyGameInstacne.h

```
// Fill out your copyright notice in the Description page of Project Settings.

#pragma once

#include "CoreMinimal.h"
#include "Engine/GameInstance.h"
#include "MyGameInstance.generated.h"

/**
 *
 */
UCLASS()
class OBJECTREFLETION_API UMyGameInstance : public UGameInstance
{
	GENERATED_BODY()

public:
	UMyGameInstance();
	virtual void Init() override;


private:
	UPROPERTY()
	FString SchoolName;
};
```

-MyGameInstance.cpp

```
#include "MyGameInstance.h"
#include "Student.h"
#include "Teacher.h"
//여기 헤더선언 부분도 주의해야할것이  언리얼 오브젝트에 선언된 MynameInstance가 가자 위에 있어야 한다.

UMyGameInstance::UMyGameInstance()
{
	SchoolName = TEXT("기본학교");
	//이 기본값은 CDO라고 하는 탬플릿 객체에 저장이 되어있음

}

void UMyGameInstance::Init()
{
	Super::Init();
	UE_LOG(LogTemp, Log, TEXT("====================="));

	SchoolName = TEXT("청강문화산업대학교");
	//기본 객체와 무관하게 생성된 MyGameInstance에는 "청강문화산업대학교"라는 학교이름이 설정 됨.

	UE_LOG(LogTemp, Log, TEXT("학교 이름 : %s"), *SchoolName);
	UE_LOG(LogTemp, Log, TEXT("학교 이름 기본값 : %s"), *GetClass()->GetDefaultObject<UMyGameInstance>()->SchoolName);
	/*
	이상태로 컴파일 하면 글자가 안나올수도 있는데
	CDO는 이 에디터가 활성화 되기 이전에 초기화 되는 순서를 가지고 있기 때문에
	에디터에서 인지를 못하는 경우가 종종있다.
	따라서 CDO를 고쳐주는 (기본값을 고쳐주는) 생성자 코드를 고치는 경우에는
	에디터를 꺼줘야 한다.!
	*/
	UE_LOG(LogTemp, Log, TEXT("====================="));


	//이부분부터는 student와 teacher를 선언하고 getter와 setter 방식을 다룰것임.
	UStudent* Student = NewObject<UStudent>();
	UTeacher* Teacher = NewObject<UTeacher>();



	//방식1 (getter와 setter)
```

```
	Student->SetName(TEXT("학생1"));
	UE_LOG(LogTemp, Log, TEXT("새로운 학생 이름 %s"), *Student->GetName());

	//방식2 (리플렉션 사용하기)
	FString CurrentTecherName; //이름 속성을 가져오기 위한 지정
	FString NewTeacherName(TEXT("이득우"));
	FProperty* NameProp = UTeacher::StaticClass()->FindPropertyByName(TEXT("Name"));
	/*
	* UTeacher의 static클래스 정보를 얻어오고 FindPropertyByName이라고 클래스에서 제공하는 함수 사용
	* FindPropertyByName은 속성 이름을 검색하여 프로퍼티에 대한 포인터를 가져오는 방식임
	* 실제로 있게 된다면 NULL이 아니게 됨.
	*/
	if (NameProp)
	{
		NameProp->GetValue_InContainer(Teacher, &CurrentTecherName);
		/*
		* MameProp은 결국 클래스에 대한 속성이다.
		* 이 속성에 대해서 우리가 지정한 인스턴스의 값을 빼올수가 있다.
		* 속성에서 인스턴스 빼오는 함수==>GetValue_InContainer();
		* Teacher가 가지는 객체의 속성중에서 이름 속성값을 가져와야하는데
		* 우리가 지정을 해줘야 한다. ==>FString CurrentTecherName 선언후 인자에 포인터로 넣기.
		*/
		UE_LOG(LogTemp, Log, TEXT("현재 선생님 이름 %s"), *CurrentTecherName);

		//이부분부터는 선생님 이름을 바꿔볼거임.

		NameProp->SetValue_InContainer(Teacher, &NewTeacherName);
		UE_LOG(LogTemp, Log, TEXT("새로운 선생님 이름 %s"), *Teacher->GetName());
	}

	//방식3(리플렉션 사용하여 함수 출력하기)
	UE_LOG(LogTemp, Log, TEXT("====================="));

	Student->DoLesson();
	//이거는 일반적인 방식

	//이 밑에서부터 리플렉션을 이용하여 선생님의 함수를 불러볼거임.
```

```
	UFunction* DoLessonFunc = Teacher->GetClass()->FindFunctionByName(TEXT("DoLesson"));
	if (DoLessonFunc)
	{
		Teacher->ProcessEvent(DoLessonFunc, nullptr);
		//인스턴스를 지정하여 ProcessEvent라는 함수를 사용하여 함수 포인터를 넘겨줌으로써 실행 가능하다.
	}

	UE_LOG(LogTemp, Log, TEXT("====================="));
}
```

-Person.h

```
// Fill out your copyright notice in the Description page of Project Settings.

#pragma once

#include "CoreMinimal.h"
#include "UObject/NoExportTypes.h"
#include "Person.generated.h"

/**
 *
 */
UCLASS()
class OBJECTREFLETION_API UPerson : public UObject
{
	GENERATED_BODY()
public:
	UPerson(); //기본값 설정하기 위핸 생성자 코드 추가.

	UFUNCTION()
	virtual void DoLesson();

	const FString& GetName() const;
	void SetName(const FString& InName);

	//외부에서 프로퍼티의 코드로 접근하기 위해 getter와 setter 선언부임
	//const로 레퍼런스를 반환하도록 하고 변경할것이 아니니깐 const 지시자를 명확하게
쓰자.

protected: //앞으로 스튜던트와 티처가 상속받을 속성을 만들것임.
	UPROPERTY()
	FString Name;

	UPROPERTY()
	int32 Year;

private:

};
```

-Person.cpp

```
// Fill out your copyright notice in the Description page of Project Settings.
```

```cpp
#include "Person.h"

UPerson::UPerson()
{
	Name = TEXT("홍길동");
	Year = 1;
}

void UPerson::DoLesson()
{
	UE_LOG(LogTemp, Log, TEXT("%s님이 수업에 참여합니다."), *Name);
}

const FString& UPerson::GetName() const
{
	return Name;
}

void UPerson::SetName(const FString& InName)
{
	Name = InName;
}
```

-Student.h

```
// Fill out your copyright notice in the Description page of Project Settings.

#pragma once

#include "CoreMinimal.h"
#include "UObject/NoExportTypes.h"
#include "Person.h"
#include "Student.generated.h"

/**
 *
 */
UCLASS()
class OBJECTREFLETION_API UStudent : public UPerson
{
	GENERATED_BODY()

public:
	UStudent();
	virtual void DoLesson() override;

private:
	UPROPERTY();
	int32 Id;

};
/*
*person을 상속받아야 하는데 person을 상속받을려면 헤더파일을 include해야 한다.
*언리얼 오브젝트 헤더에서 다른 헤더를 include하게 되면 조심해야 하는데언리얼 헤더툴에서 generated.h 파일 밑에 include 했기 때문이다.
*따라서 언리얼 오브젝트에서 선언할때는 generated.h가 가장 밑에 있는것이기본 규칙이다.
*/

/*
* Person에서 구현된 DoLesson 함수를 실행한 후에 Student의 DoLesson을 진행할 것임.
*/
```

-Student.cpp

```
// Fill out your copyright notice in the Description page of Project Settings.


#include "Student.h"

UStudent::UStudent()
{
	Name = TEXT("이학생");
	Year = 1;
	Id = 1;
}

void UStudent::DoLesson()
{
/*
* Person에서 구현된 DoLesson 함수를 실행한 후에 Student의 DoLesson을 진행할 것임.
*/
	Super::DoLesson();

	UE_LOG(LogTemp, Log, TEXT("%d학년 %d번 %s님이 수업을 듣습니다."), Year, Id, *Name);

}
```

-Teacher.h

```
// Fill out your copyright notice in the Description page of Project Settings.

#pragma once

#include "CoreMinimal.h"
#include "UObject/NoExportTypes.h"
#include "Person.h"
#include "Teacher.generated.h"

/**
 *
 */
UCLASS()
class OBJECTREFLETION_API UTeacher : public UPerson
{
	GENERATED_BODY()

public:
	UTeacher();
	virtual void DoLesson() override;

private:
	UPROPERTY()
	int32 Id;


};
```

-Teacher.cpp

```
// Fill out your copyright notice in the Description page of Project Settings.


#include "Teacher.h"

UTeacher::UTeacher()
{
	Name = TEXT("이선생");
	Year = 3;
	Id = 1;
}

void UTeacher::DoLesson()
{
	Super::DoLesson();
	// 여기서 만약 DoLesson()에 대해서 자동완성이 안된다면 컴파일이 반영 안된 거임
	// 헤더파일을 리프레시(줄바꿈, 띄어쓰기 등등 한 후에 컴파일 하면 generated.h가 재생성 되면서 상위클래스 정보를 얻을 수 있을거임
	UE_LOG(LogTemp, Log, TEXT("%d년차 선생님 %s님이 수업을 강의하십니다."),Year, *Name);
}
```

**정리

언리얼 리플렉션 시스템의 활용

1. 리플렉션 시스템을 사용해 언리얼 오브젝트의 특정 속성과 함수를 이름으로 검색할 수 있다.
2. 리플렉션 시스템을 사용해 접근 지시자와 무관하게 값을 설정할 수 있다.
3. 리플렉션 시스템을 사용해 언리얼 오브젝트의 함수를 호출할 수 있다.

언리얼 엔진의 기본 프레임웍은 리플렉션을 활용해 구축되어 있으므로
언리얼 엔진을 이해하기 위해서는 리플렉션 시스템을 이해하는 것이 필요함.

언리얼 오브젝트의 특정한 속성과 함수를