1. 정적 바인딩 VS 동적 바인딩

<https://gamelaboratory.tistory.com/entry/CC-%EC%A0%95%EC%A0%81-%EB%B0%94%EC%9D%B8%EB%94%A9%EA%B3%BC-%EB%8F%99%EC%A0%81-%EB%B0%94%EC%9D%B8%EB%94%A9>

바인딩(binding)이란 프로그램에 사용된 구성 요소의 실제 값 또는 프로퍼티를 결정짓는 행위를 의미합니다.

-정적 바인딩은 컴파일 시간에 바인딩이 되는것이고 동적 바인딩은 실행 시간에 바인딩이 되는 것입니다. 동적 바인딩을 사용하는 이유는 실행 도중에 필요한 객체의 함수를 호출함으로써 유연성을 가질 수 있습니다.

1. List와 Vector 차이

<https://jaehogame.tistory.com/entry/STL-Vector%EC%99%80-List-%EC%B0%A8%EC%9D%B4%EC%A0%90>

-vector는 미리 일정 크기의 메모리를 정해놓고 그 이상값이 추가되면 더 큰 메모리를 할당하는 반면에 list는 값을 넣을 때마다 메모리를 할당합니다. vector는 random access가 가능하여서 search는 빠르지만 중간 데이터의 삽입 삭제에는 성능이 좋지 않기 때문에 중간에 삽입 삭제가 빈번한 경우에는 list를 사용하는 것이 좋습니다.

1. hashtable , 충돌시 해결 방법은? <https://velog.io/@cyranocoding/Hash-Hashing-Hash-Table%ED%95%B4%EC%8B%9C-%ED%95%B4%EC%8B%B1-%ED%95%B4%EC%8B%9C%ED%85%8C%EC%9D%B4%EB%B8%94-%EC%9E%90%EB%A3%8C%EA%B5%AC%EC%A1%B0%EC%9D%98-%EC%9D%B4%ED%95%B4-6ijyonph6o>

-해시 테이블은 연관배열 구조를 이용하여 Key값과 대응되는 value를 저장하는 자료구조입니다. Hashing은 Hash 함수를 이용하여 Key를 Hash Key로 변경하는 것입니다.

충돌이 나면 2가지 방법이 있는데 Open Adress방식으로 빈 HashTable공간을 찾거나

연결 List를 이용하여 Chaning방법을 사용합니다.

1. quicksort와 MergeSort? 구현 방법을 말로 설명

<https://gmlwjd9405.github.io/2018/05/10/algorithm-quick-sort.html>

<https://gmlwjd9405.github.io/2018/05/08/algorithm-merge-sort.html>

퀵소트 – 일반적으로 가장 빠른 알고리즘 리스트에 있는 한 요소를 선택하고 이것을 피벗이라고 하고 피벗을 중심으로 작은 요소들을 왼쪽 큰 요소들을 오른쪽으로 옮기고 이런 분할 과정을 나누지 못할 때 까지 반복합니다.

머지소트 – 배열을 같은 크기의 2개의 부분 배열로 분할하고 더 이상 나눌 수 없을 때까지 분할합니다. 그 후 새로운 배열을 만들고 2개의 리스트의 값들을 처음부터 하나씩 비교하여 두 리스트 값중 더 작은값을 새로운 리스트로 옮기고 하나의 리스트가 끝나게 되면 나머지 값들은 전부 리스트로 옮기는 과정을 거쳐 정렬을 합니다.

1. 메모리 영역에 대해서 설명 (데이터, 코드, 스택, 힙)

<http://tcpschool.com/c/c_memory_structure>

코드 영역 – 실행할 프로그램 코드가 저장되는 영역

데이터 영역 – 전역 변수와 정적 변수가 저장되는 영역

스택 영역 – 함수의 호출과 관계되는 지역변수와 매개변수가 저장되는 영역

힙 영역 – 프로그래머가 직접 할당/해제하는 메모리 공간으로 동적할당을 하는 메모리 영역입니다.

1. AVL트리란 무엇인가?

<https://www.zerocho.com/category/Algorithm/post/583cacb648a7340018ac73f1>

VAL트리란 자가 균형 이진 트리로 좌측의 서브트리와 우측의 서브트리의 높이차이가 1이하로 이루어진 자료구조입니다.

1. virtual 키워드와 virtual table

<http://tcpschool.com/cpp/cpp_polymorphism_virtual>

virtual 키워드는 파생 클래스에서 재정의 할 것으로 기대하는 멤버 함수를 정의하는데 사용되는 키워드로 이러한 가상 함수를 가지고 있는 클래스는 가상 함수 테이블을 가지고 가상함수를 호출하면 가상함수 테이블에 접근하여 자신이 필요한 함수의 주소를 찾아 호출하게 되는 매커니즘 입니다.

virtual 키워드 소멸자는 기반클래스 인스턴스가 자식클래스를 동적으로 할당됐을 때 virtual 소멸자를 사용하지 않으면 자식클래스의 소멸자가 호출되지 않을 수 있는 것을 방지하기 위해 사용합니다.

1. new와 malloc의 차이

<https://drunkenpsycho.tistory.com/13>

new는 키워드고 malloc은 함수이기 때문에 헤더파일을 포함해야하고 리턴타입이 void형이므로 정확한 메모리크기와 자료형을 지정해놔야 한다. 또 new는 생성자가 자동으로 호출되고 생성과 동시에 초기화가 가능하지만 malloc은 불가능하다.

1. 지역변수와 전역변수가 메모리상에 어디에 놓이나요

-지역변수는 Stack영역에 놓이고 전역변수는 데이터 영역에 놓입니다.

1. 컨택트 스위칭에 대해 설명하시오

<https://nesoy.github.io/articles/2018-11/Context-Switching>

-현재 진행하고 있는 Task(Process,Thread)의 상태를 저장하고 다음 진행할 Task의 상태 값을 읽어오는 과정을 말합니다.

1. 스레드와 프로세스 차이점

-프로세스는 현재 실행중인 프로그램을 프로세스라 하며 이런 프로세스를 여러 태스크로 나나 작업하는 것을 스레드라고 합니다. 프로세스와 스레드의 차이는 프로세스와 달리 스레드는 스택영역을 제외한 나머지 영역을 공유합니다. 이 때문에 컨텍트 스위칭이 더 빠른 효과를 얻습니다.

1. 메모리 단편화

메모리의 빈공간 또는 자료가 여러 개의 조각으로 나뉘는 현상으로 메모리의 빈공간을 사용하지 못하는 경우입니다.

내부 단편화 : 할당될 크기에 비해 실제 할당된 자원크기가 커서 해당 단위 자원내부에 사용하지 않는 메모리가 발생.

외부 단편화 : 분할된 영역이 할당된 프로그램의 크기보다 작아서 할당할 수 없어 모두 빈공간으로 남아있음

1. 스마트 포인터에 대해 설명해주세요.

<http://tcpschool.com/cpp/cpp_template_smartPointer>

-C++에서 new키워드를 사용하여 동적으로 할당받은 메모리를 반드시 delete키워드를 사용하여 메모리를 해제시켜줘야 메모리 누수가 발생하지 않는데, 이러한 메모리 누수로 부터 프로그램의 안정성을 보장하기 위한 것이 스마트포인터 입니다. 스마트 포인터는 unique, shared, weak로 나눠져 있는데 unique\_ptr은 하나의 스마트 포인터만이 특정 객체를 소유할 수 있는것이고, shared\_ptr는 하나의 특정 객체를 참조하는 포인터가 몇 개인지를 참조하는 포인터로 참조가 0이 되면 메모리를 자동으로 해제해주고 weak\_ptr는 하나 이상의 shared\_ptr인스턴스가 소유하는 객체에 대한 접근을 제공하지만 소유자의 수에 포함되지 않도록 하는 포인터 입니다.

1. STL map과 Hash Map(Unorded\_map)의 차이

<https://gracefulprograming.tistory.com/3>

STL map은 기본적으로 Red-Black트리로 모든 데이터는 정렬되어 저장되고

Hash Map은 hash-table을 사용하여 데이터를 저장합니다.

1. TCP와 UDP차이

<https://swalloow.tistory.com/77>

TCP는 신뢰성있는 데이터 전송을 지원하는 연결지향형 프로토콜이고 UDP는 비 연결형 입니다. 때문에 TCP는 3-way handshaking이라는 과정을 통해 연결한 후 통신을 시작하고

데이터의 순서나 오류를 제어하느라 속도가 느린반면 UDP는 속도가 빠릅니다.

1. 뮤텍스 세마포어 차이

<https://jwprogramming.tistory.com/13>

<https://worthpreading.tistory.com/90>

세마포어는 공유된 자원의 데이터를 여러 프로세스가 접근하는 것을 막는것이고 뮤텍스는 여러 쓰레드가 접근하는 것을 막는 것입니다.

1. 추상클래스와 interface차이

하나 이상의 순수가상함수를 포함하는 클래스를 추상클래스라고 하며 모든 함수들이 순순수가상함수 이루어진 것을 인터페이스라 합니다.

1. 스택과 큐 차이점

<https://winplz.tistory.com/entry/%EC%9E%90%EB%A3%8C%EA%B5%AC%EC%A1%B0%EC%8A%A4%ED%83%9D%EA%B3%BC-%ED%81%90-stack-and-queue>

스택은 후입선출 구조로 마지막에 삽입된 원소를 삭제/삽입하고 큐는 선입선출 구조로 맨 처음 삽입된 원소를 삽입/삭제 하는 구조입니다.

1. delegate란?

c++의 함수 포인터에 대응되는 타입으로 함수를 포인터처럼 메스드를 안전하게 캡슐화하는 방식입니다.

즉 A클래스 메서드를 B클래스로 넘겨 B클래스에서 A클래스의 메서드를 호출하게 해주는역할을 하는 (A의 행위를 B에게 위임하여 B가 A의 행위를 대신하는 대리자).

1. GarbigeCollector에 대해서 설명

<http://egloos.zum.com/sweeper/v/3205731>

프로그램에서 더 이상 사용되지 않는 쓸모없는 데이터를 자동으로 정리해주는 프로그램입니다. 언리얼은 Garbage collection을 수행하기 위해 리플렉션 시스템을 사용하는데 엔진이 객체와 속성들을 알고 있고 그러기에 더 이상 사용되지 않아 삭제되어도 좋은 객체들을 구분할 수 있기 때문이다 GC는 기본적으로 시스템이 알아서 수행하기 때문에, 개발자의 희망 시점과는 무관하게 실행된다.

게임이 진행되는 도중보다 가급적 로딩 타임에 가비지가 수집되면 좋기에 로딩 시점에 한번씩 강제로 실행해주는 것도 나쁘지 않을 수 있다.

1. Call by Value Vs Call by Reference

<https://wayhome25.github.io/cs/2017/04/11/cs-13/>

-Call by value는 들어온 매개변수를 복사하여 함수의 인자로 전달 하여서 함수안의 인자가 변경되어도 외부의 변수값은 병경되지 않지만 call by reference는 인자로 전될다는 변수의 레퍼런스의 인자를 가져오기 때문에 함수안에서 인자가 변경되면 전달된 객체의 값도 함께 변경됩니다.

1. override와 overload의 차이

override – 상위 클래스의 함수를 재정의

orverload – 하나의 함수 이름으로 여러 개의 매개변수를 받아 각각에 맞는 함수를 만듦.

같은 클래스에서 같은 이름을 가진 메소드가 여러개가 존재하고 있는 상태이다.

1. static\_cast, dynamic\_cast, reinterPret\_cast

<http://egloos.zum.com/sweeper/v/1907485>

reinterPret\_cast : 어떠한 포인터 타입도 강제로 변환해줌

static\_cast : 형변환에 대한 타입체크를 run-time에 하지않고, compile 타임에 정적으로 수행한다.

dynamic\_cast : dynamic\_cast는 **런타임**에 (동적으로) 상속 계층 관계를 가로지르거나 다운캐스팅시 사용되는 캐스팅 연산자이다.

캐스팅의 실패는 NULL(포인터)이거나 예외(참조자)를 보고 판별할 수 있다

dynamic\_cast는 런타임에 **다형성**을 이용하여 모호한 타입 캐스팅을 시도할 때 (다형성 위배),

엉뚱한 변환 결과가 넘어가지 않도록 하여, 런타임 오류가 방지하는 역할을 한다.

1. 그래픽스 파이프라인에 대해 설명해주세요.

3차원 이미지를 2차원 래스터 이미지로 표현하기 위한 단계로 래스터(raster)란 컴퓨터에서 화상 정보를 표현하는 한가지 방법으로 2차원 배열의 픽셀로 구성한다.

1. C, C++ 차이점

<https://jjoreg.tistory.com/entry/C-%EA%B3%BCC%EC%9D%98-%EC%B0%A8%EC%9D%B4%EC%A0%90>

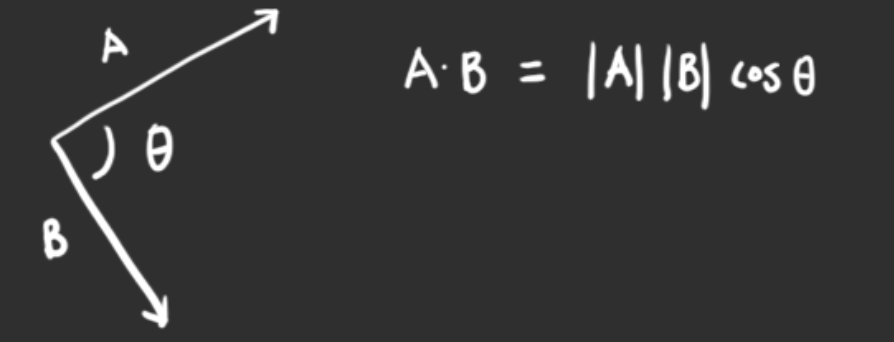
C는 절차지향적 언어 C++은 객체지향언어고 class를 사용하여 객체를 중심으로 한 코딩방식 입니다. C는 하향식 접근 방법으로 순차적 수행이 되도록 서로간의 연결을 중요시하고 C++은 상향식으로 접근 합니다.

1. 그래픽스의 벡터란 무엇일까요?

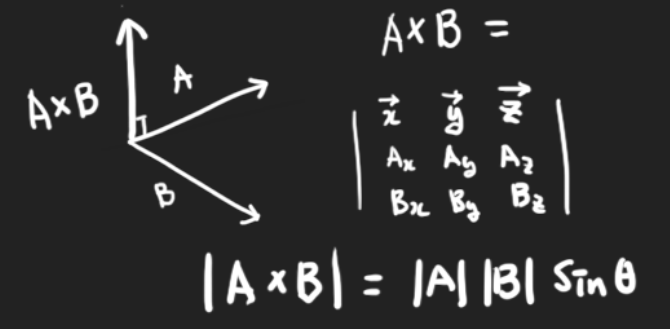
크기와 방향을 가지고 있는 물리량을 말하며 게임 프로그래밍에 있어 기본적인 단위입니다.

1. 내적과 외적이 무엇인가?

<http://rapapa.net/?p=2974>

내적 

외적

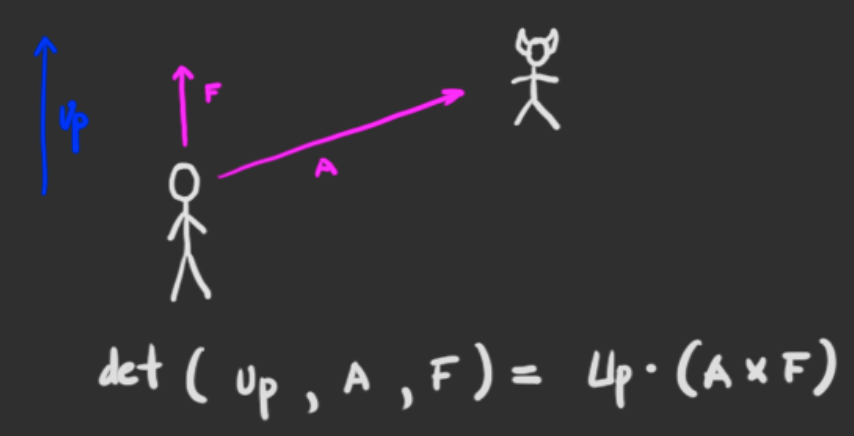


1. 적이 앞쪽에 있는지 어떻게 판단할까요?



내 앞방향 벡터와 나로부터 적까지의 벡터와 내적 시 결과값이 양수면 앞 음수면 뒤

1. 적이 왼쪽인지 오른쪽인지 어떻게 판단할까요



업벡터와 내적 (앞방향 벡터와 적방향 벡터를 외적) 이 0보다 크면 오른쪽 작으면 왼쪽

1. 랜더링파이프라인 과정을 얘기해주세요.

로컬스페이스 -> 월드 스페이스 -> 뷰 스페이스-> 절두체 컬링(후면추려내기) -> 조명 -> 클리핑 -> 투영-> 뷰포트 변환 -> 래스터 라이즈

1. 데드락이 뭔가요?

<https://jwprogramming.tistory.com/12>

교착상태라고 하며 여러 프로세스가 한정된 자원을 사용할 때 필요한 자원을 얻지 못해 무한정 대기하는 것을 데드락이라고 합니다.

1. 동기화란 무엇인가요?

https://arcsit.tis tory.com/entry/Thread-%EB%8F%99%EC%8B%9C%EC%84%B1-%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%98%EB%B0%8D-%EB%8F%99%EA%B8%B0%ED%99%94

멀티스레드에서 하나의 자원을 여러 스레드가 공유할 때 동시에 자원에 접근할 경우 생기는 문제점을 보완하기 위한 공유 매커니즘

1. 리플렉션이란?

<https://www.unrealengine.com/ko/blog/unreal-property-system-reflection?lang=ko>

리플렉션(Reflection)은 프로그램이 실행시간에 자기 자신을 조사하는 기능

1. template 키워드

<https://blockdmask.tistory.com/43>

함수나 클래스를 개별적으로 다시 작성하지 않아도, 여러 자료형으로 사용할 수 있도록 하게 만들어놓은 키워드 입니다.

1. 가장 자신있는 언어는 무엇인가요?

C++이 가장 자신있고 오래 사용하였습니다.

1. 토폴로지소트와 부르트포스 알고리즘

<https://gmlwjd9405.github.io/2018/08/27/algorithm-topological-sort.html>

토폴로지소트 : 방향 그래프에 존재하는 각 정점들의 선행 순서를 위배하지 않으면서 모든 정점을 나열하는것

<https://steemit.com/kr-dev/@gyeryak/easyalgo-2-bruteforce>

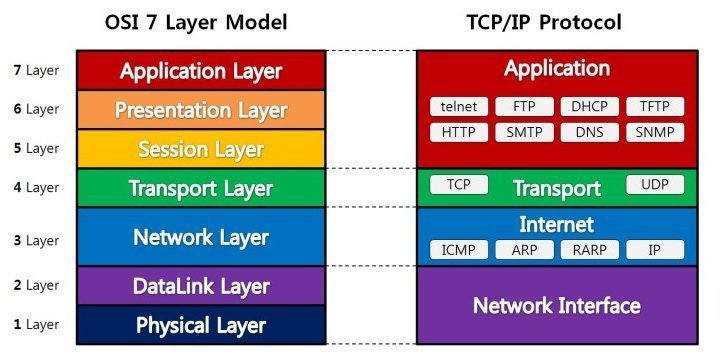
브루트포스(짐승+힘) 알고리즘 : 문제를 해결하기 위해서 가능한 모든 경우에 대해 모두 직접 해 보는 방법.

1. OSI-7계층

<https://shlee0882.tistory.com/110>

-네트워크에서 통신이 일어나는 과정을 7단계로 나눈 것을 말함.

7계층으로 나눈 이유 : 통신이 일어나는 과정을 단계별로 파악할 수 있기 때문이다.



1. 싱글톤 패턴에 대해 설명해주세요.

-애플리케이션이 시작될 때 어떤 클래스가 최초 한번만 메모리를 할당하고 그 메모리에 인스턴스를 만들어 사용하는 디자인 패턴으로 전역적으로 단 하나의 인스턴스만 사용하는 패턴입니다.

사용 이유 : 전역 인스턴스이기 때문에 다른 클래스의 인스턴스들이 데이터를 공유하기 쉬움

문제점 : 싱글턴 인스턴스가 너무 많은 일을 하거나 많은 데이터를 공유시킬 경우 다른 클래스의 인스턴스들 간에 결합도가 높아져서 객체 지향 설계 원칙에 어긋날 수 있고 에러를 범할 수 있습니다. 또 멀티스레드 환경에서 동기화 처리를 안하면 인스턴스가 두개가 생성될 수 있다.

1. 디자인패턴에 대해 공부해보셨나요? 그러면 알고계신거에 대해 설명해주세요.

컴퍼넌트 패턴

싱글턴 패턴

옵저버 패턴

1. OOP에 대해 설명해주세요

<https://goodgid.github.io/What-is-OOP/>

Object-Oriented Programmibng으로 객체 지향적 프로그래밍이다. 데이터와 이를 처리하는 함수들을 하나의 독립된 객체로 동작하고, 재사용을 사용한다.

1. 추상화

-대상의 특성 중 불필요한 부분을 무시하고 필요한 공통점만 다룬다

-현실의 복잡성을 극복하고 문제에 집중한다

2. 캡슐화

-상태와 행위의 캡슐화, 외부에서 객체이ㅡ 상태를 마음대로 변경할 수 없도록 보호

3. 상속성

- 부모 클래스의 속성과 기능을 상속받아 동일하게 사용, 반복적인 행위를 없애줌

4. 다형성

-동일 요청에 대해 서로 다른 방식으로 응답할 수 있도록 만드는 것 (오버라이딩, 오버로딩)

1. Virtual void test() = 0;

순수 가상 함수로 파생 클래스는 반드시 함수를 재정의 해야만 인스턴스를 생성할 수 있음, 모든 함수들이 순수가상함수로 이루어진 클래스를 인터페이스라함.

1. 평면과 벡터가 충돌했는지 안했는지 어떻게 판별할가요?

<https://thrillfighter.tistory.com/191>

1. Effective C++ 다뤄보신적 있나요? 알고 계신가요?

Smart Pointer, auto, template

1. 메모리 계층에 대해서 아시나요?

캐시 -> 레지스터 -> 메모리 -> 하드

1. BST 평균, Worst, Worst Case 설명
2. 정렬 알고리즘 알고 계신거 말씀해주세요.
3. 힙에 대해서 설명하고 삽입삭제를 과정을 말씀해주세요.
4. 전위 중위 후위에 대해서 말로 설명해주시겠어요?
5. static에 대해서 설명해주시겠어요?
6. Lvalue Rvalue 설명해보세요
7. at과 []의 차이
8. 52. 스택 오버플로우가 왜 일어나는가?
9. 53. 메모리 누수가 무엇인가? 막기위해서는 어떻게 해야하는가?
10. 54. 콜 스택이 무엇인지?
11. 55. 멀티프로세스나 멀티스레드를 사용해보신적 있나요?

예 프로젝트에서 스레드를 사용해서 틱카운터를 이용해서 현재 시간과 이전시간의 차이를 구해 delta값을 사용하여서 FPS를 구하고 씬을 업데이트 하는 기능을 사용해봤습니다.

1. 56, 람다식이란 무엇인가?
2. 57. 프레임워크와 라이브러리의 차이를 말해주세요.
3. 58. 디버깅 경험이 있으신가요?-> 중단점 이런거 말고 메모리덤프 이런거
4. 59. 형상관리 툴(=소스컨트롤. Git, SVN, Unity Collab) 사용해보셨나요?
5. 60. 얕은 복사와 깊은 복.사의 차이점을 아시나요?
6. 61. 참조와 포인터의 차이 (call by reference)
7. 62 업캐스팅, 다운캐스팅 설명
8. 63 정규화
9. 64. 3-hand shaking이 뭔지 설명해주세요.
10. 65. 최적화를 위해서 어떤것을 해보셨나요?
11. 66 생성자함수 중 초기화리스트와 대입의 차이
12. 67 멀티쓰레드의 장점이 무엇이고 왜 써야되나요?

<https://m.blog.naver.com/PostView.nhn?blogId=yuc1206&logNo=50141800358&proxyReferer=https%3A%2F%2Fwww.google.com%2F>

1. 68. 제네릭프로그래밍이 무엇인지 아시나요?(템플릿)
2. 69. namespace 사용 이유는?
3. 70. test코드 장점과 사용해보신적 있나요?
4. 71. volatile
5. 72 다익스트라 알고리즘
6. 73.밸만포드 알고리즘과 다익스트라 알고리즘 차이점은?(넥슨)
7. 74. 다익스트라 알고리즘은 보통 우선순위 큐를 사용하는데 이렇게 사용하는 이점은?(넥슨)
8. 75. C++과 C#의 차이 (garbage Collection)
9. 76 오퍼레이터에 대해 설명하기
10. 77 friend 키워드란?
11. 78 const란?
12. 79 vector에서 reserve랑 resize의 차이
13. 블루프린트 vs 네이티브 차이점과 장단점

<https://docs.unrealengine.com/ko/Resources/SampleGames/ARPG/BalancingBlueprintAndCPP/index.html>