2009, Copyright All rights reserved

3D Game Programming 52
-Game Engine 제작

afewhee@gmail.com

GaLic. Heesung Oh
2009, Copyright All rights reserved

- Game Engine - 개요
- Engine 제작 - 개요
- Engine 제작

GaLic. Heesung Oh
2009, Copyright All rights reserved

- 좁은 의미
  - 게임 실행 속도를 빠르게(Speedy)
  - 자원을 효율적으로(Efficient)
  - 개발을 쉽게(Easy)

- 넓은 의미
  - 게임 개발 툴킷(Game Software Development Kit)

- 기타
  - 게임에서 활용하기 위한 라이브러리 집합
  - 렌더링 엔진의 경우 단순히 OpenGL, 또는 DirectX를 Wrap하는 경우라면 커다란 의미는 없음

GaLic. Heesung Oh
2009, Copyright All rights reserved

- 통합 엔진 구성
  - 자원 관리: 그래픽, Model, 미디어, 게임 데이터 관리
  - 제작 도구: 지형 툴(Tool), 캐릭터 툴, Effect 툴
  - 렌더링:

- 특화된 엔진
  - 역학 엔진: 강체, 유체 운동에 대한 엔진
  - 인공지능: 길 찾기, 상태 머신 관리
  - 네트워크: 분산서버, P2P, C/S

- 엔진 종류
  - 통합 엔진: 퀘이크, 언리얼, 파크라이
  - 공개 엔진: Ogre, Irrlicht, Genesis
  - 부분 엔진: 토카막, FastTree, Path Finder

2009, Copyright All rights reserved

PART 2.

엔진 제작 개요

GaLic. Heesung Oh
2009, Copyright All rights reserved

- 계층별 설계
  - 소프트웨어 분석처럼 구체적인 구현은 생각하지 않고 코드와 코드 사이의 관계를 먼저 규명, 각 코드들의 독립성과 응용에 대한 등급을 정함

- 개발자 입장에서의 설계
  - 프로그래밍 전문가, 스크립터, 게임 이외 응용 프로그래머에 대한 인터페이스 필요

  - ※ 프로그램의 논리적 분리는 프로그래머의 역할 분담의 기준이 된다.

GaLic. Heesung Oh
2009, Copyright All rights reserved

Application Program

다양한 계층의 모듈을 가지고 최종적으로 게임 개발 툴 또는 다양한 게임 개발 응용 프로그램

Common Package

게임개발에 필수적인 프로그램과 라이브러리

Base Package

게임개발 외에도 사용할 수 있는 라이브러리(운영체제, 게임 라이브러리 등과 개발 플랫폼에 독립)

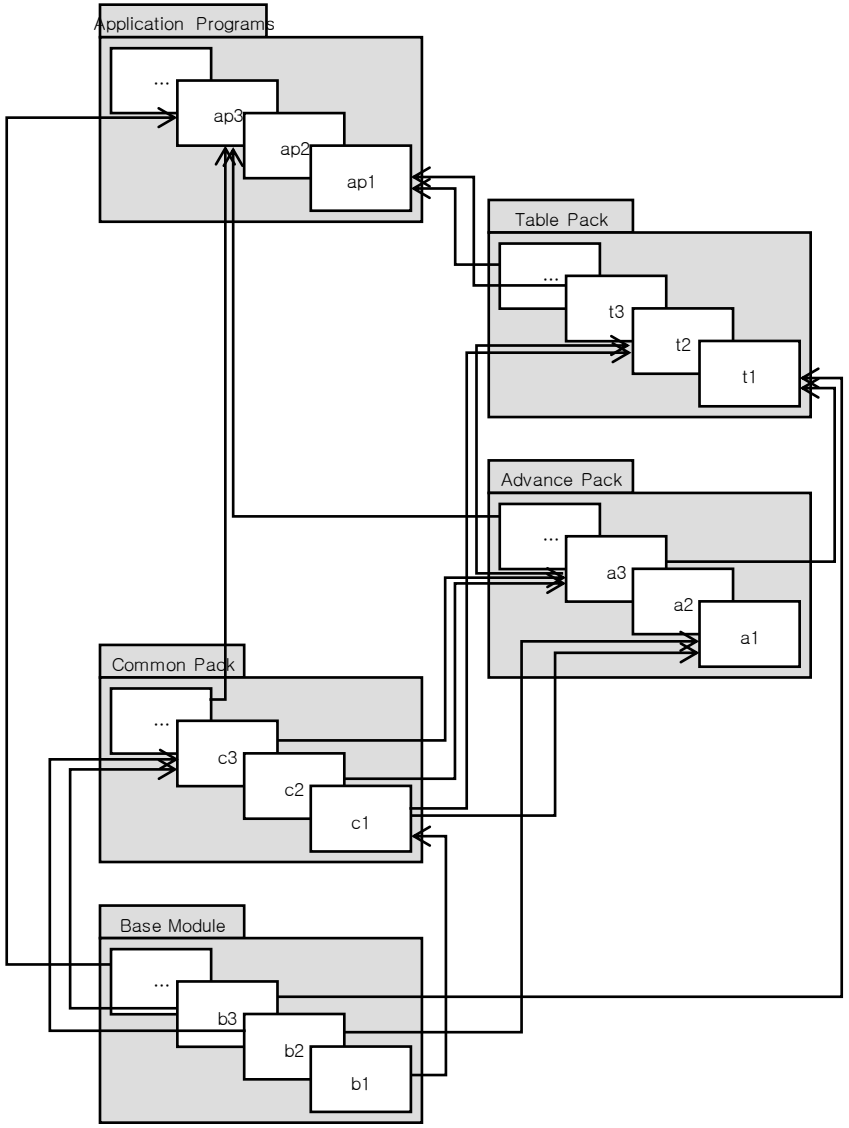


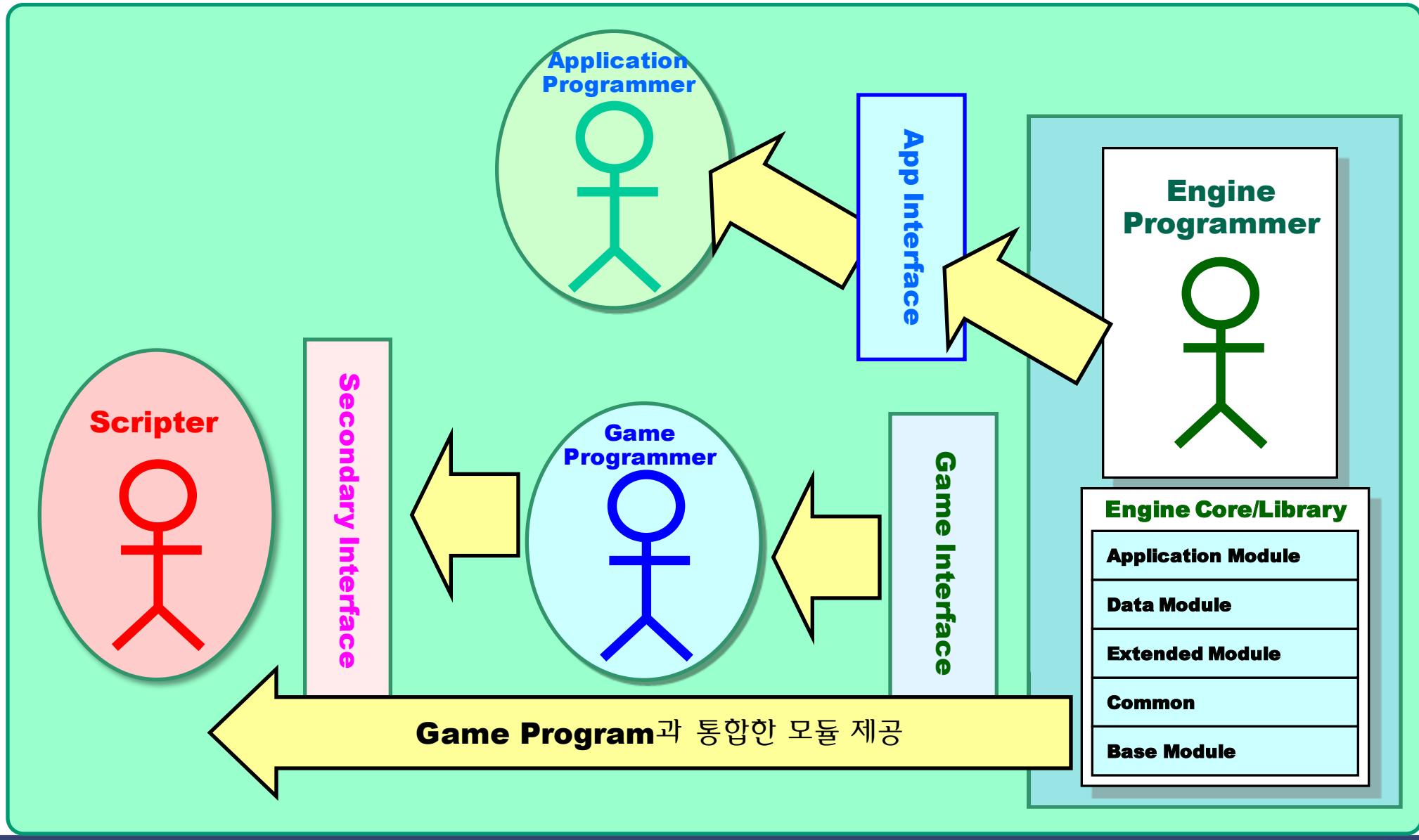
Table Package

Extended 모듈이 사용하는 게임 리소스를 관리하는 프로그램

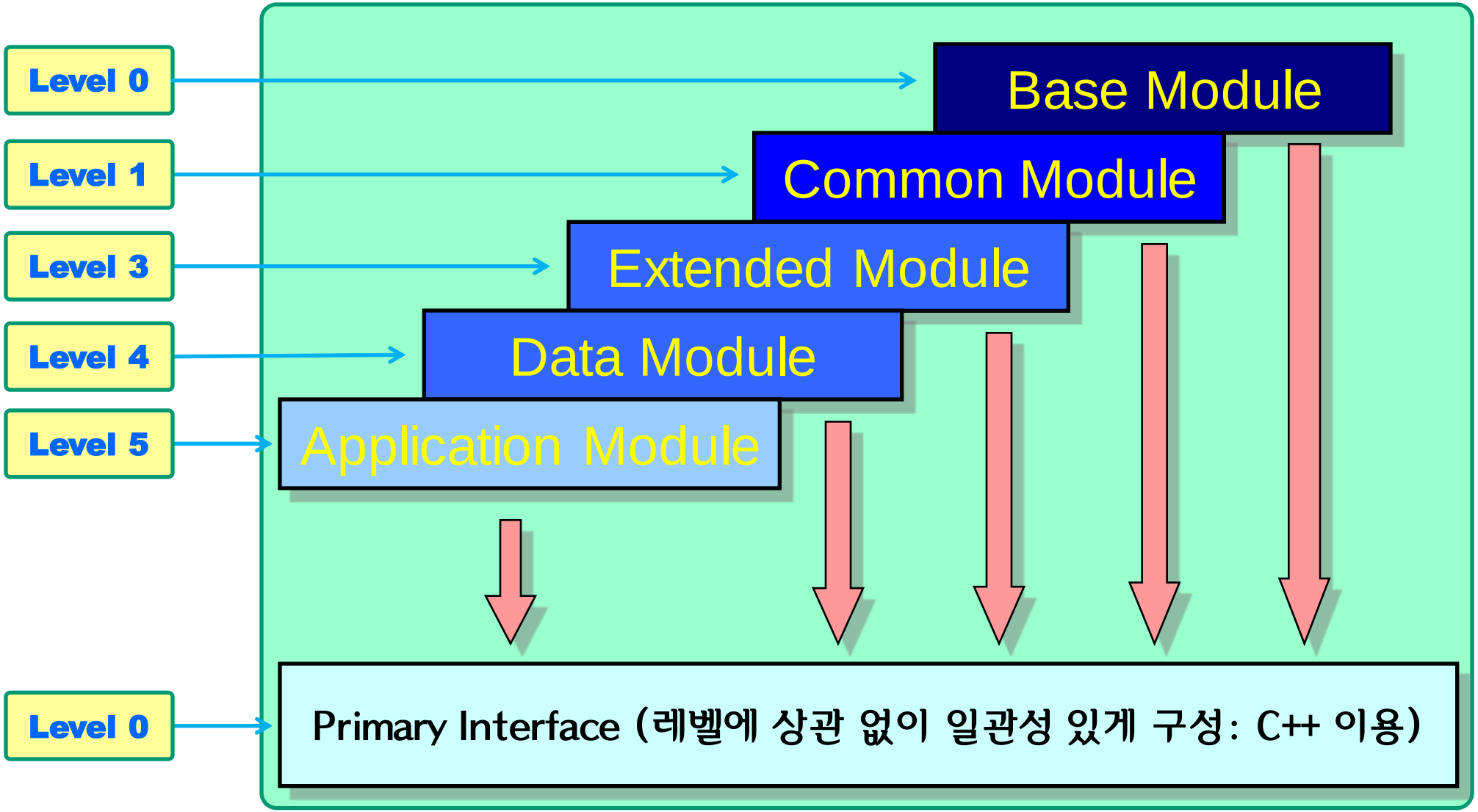
Extended Package

게임 내에서 구현되어야 할 기능들
예)지형, 모델링, 쉐이딩,라이팅…
DirectX, OpenGL 등 게임의 기본 라이브러리에 의존하는 2차 모듈

GaLic. Heesung Oh
2009, Copyright All rights reserved

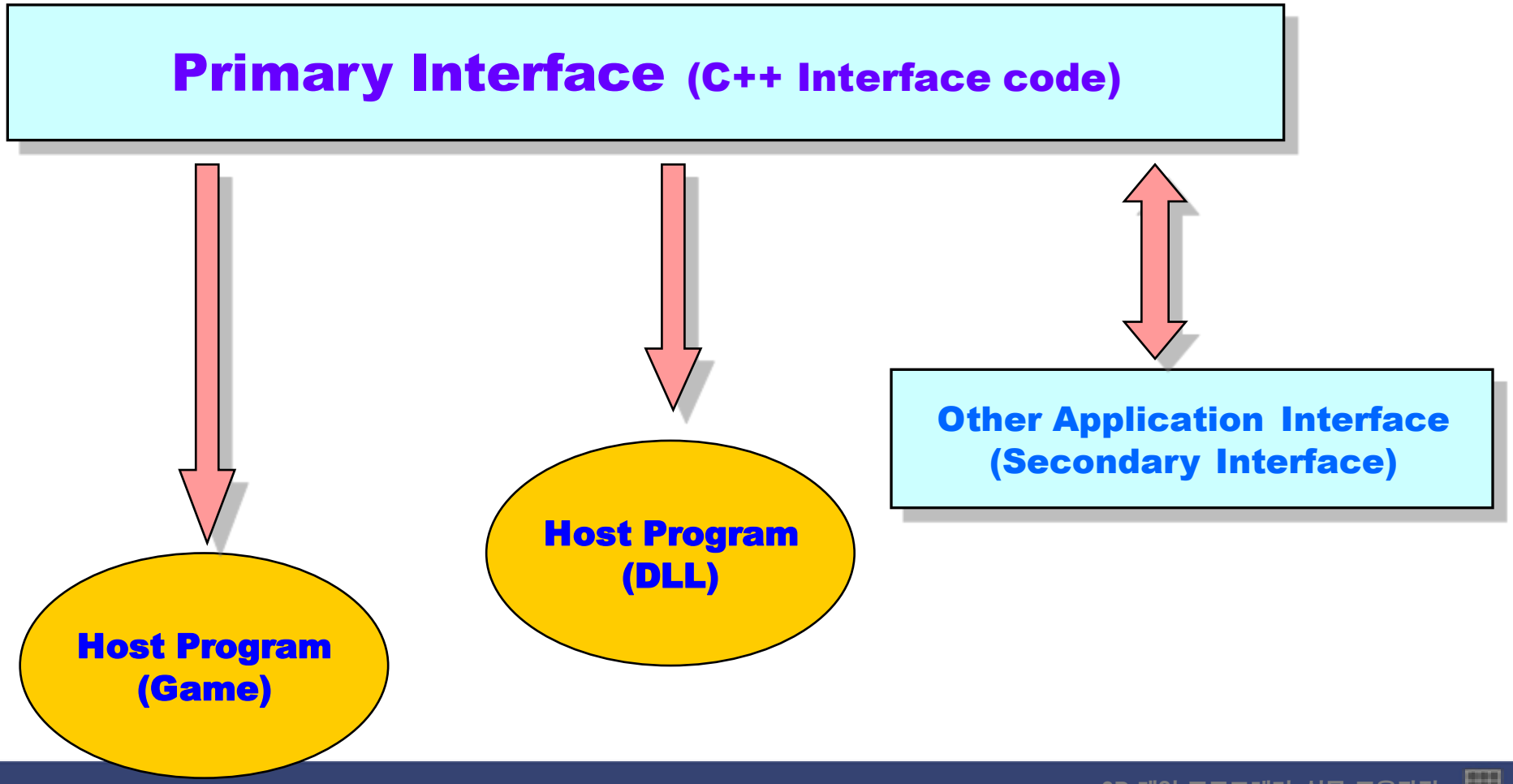


GaLic. Heesung Oh
2009, Copyright All rights reserved



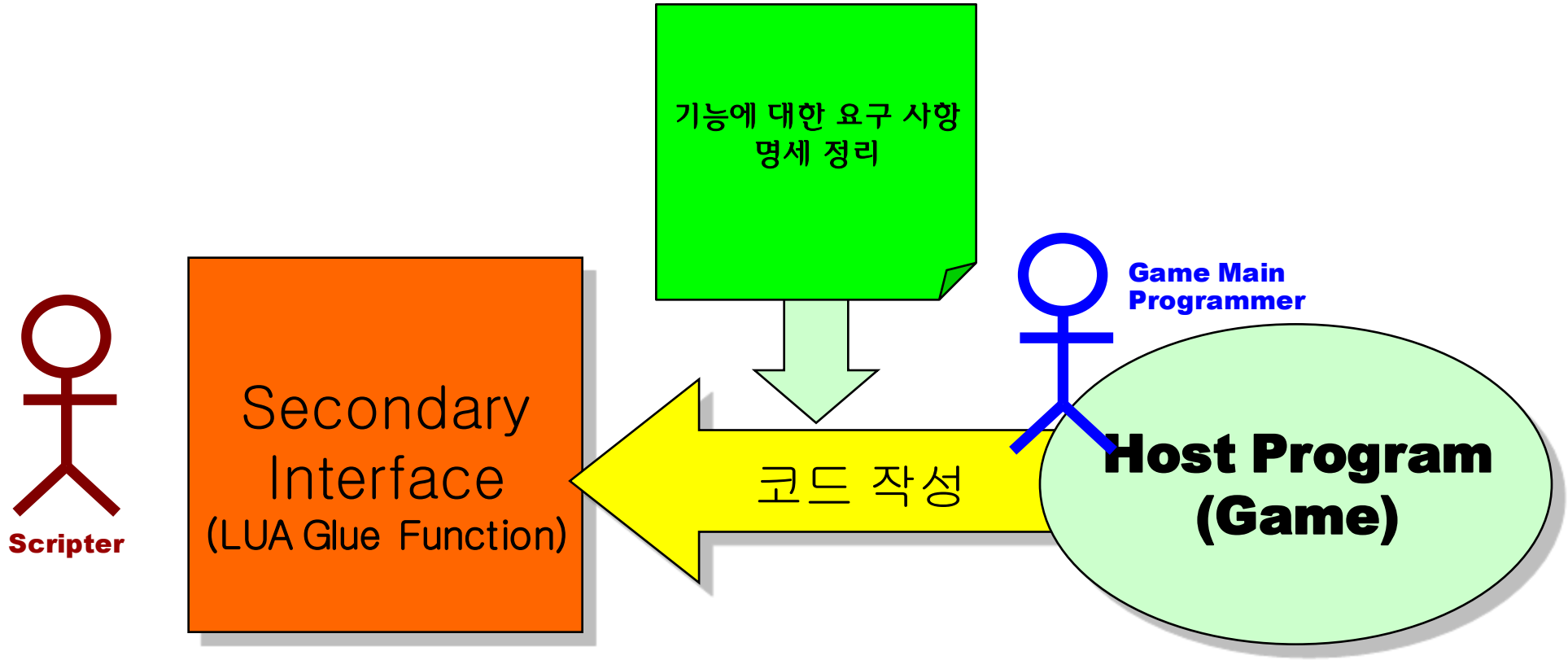
GaLic. Heesung Oh
2009, Copyright All rights reserved

- Primary Interface
  - 응용 프로그램에서 게임 엔진에 직접 접근 하는 제 1 인터페이스
  - 인터페이스는 다른 응용 프로그램과의 관계도 충분히 고려해서 설계



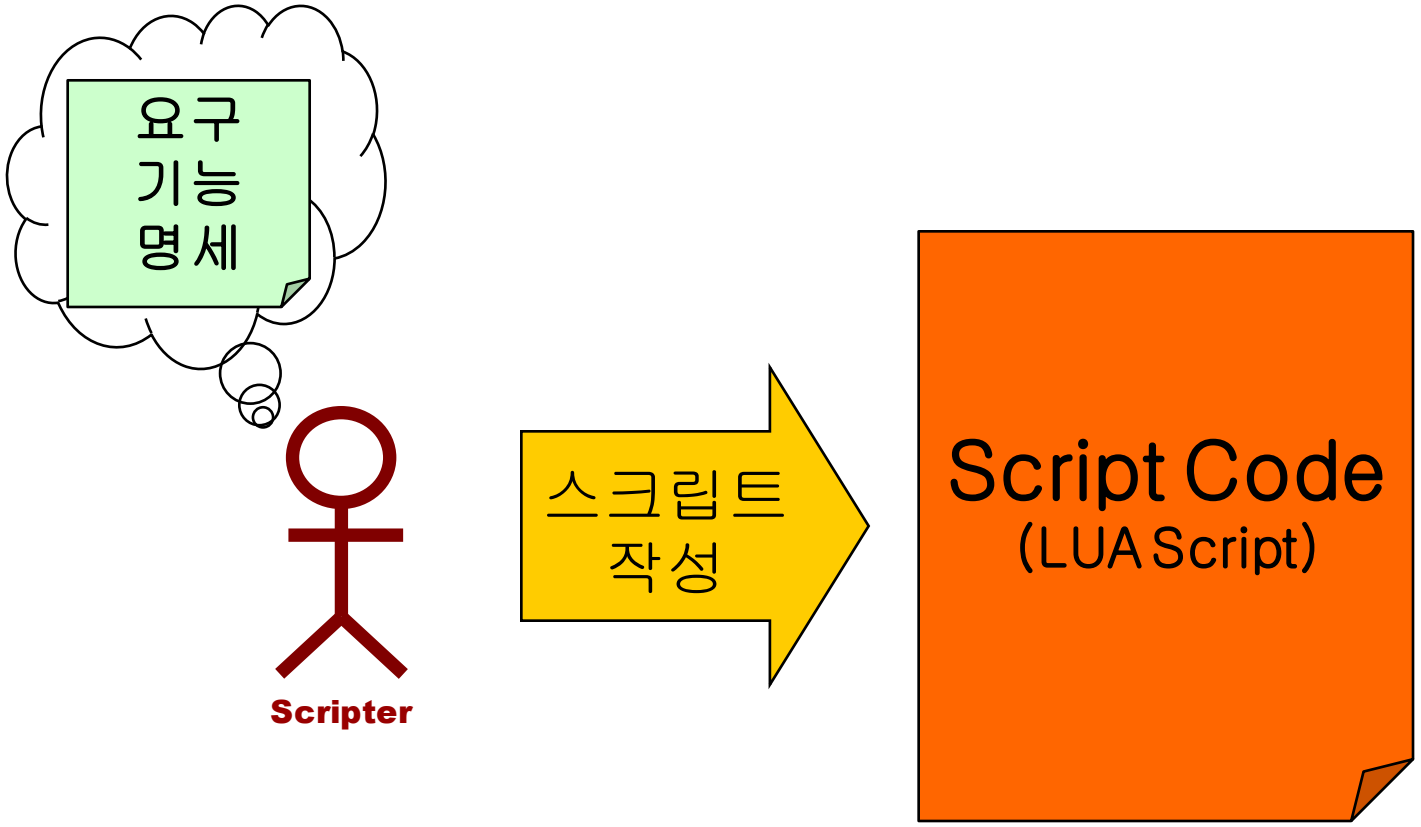
GaLic. Heesung Oh
2009, Copyright All rights reserved

- 게임 메인 프로그래머에 의한 Secondary Interface 제작
  - 스크립터가 요구한 명세서에 기술된 모든 인터페이스를 먼저 작성한 후 구체적인 코드를 작성

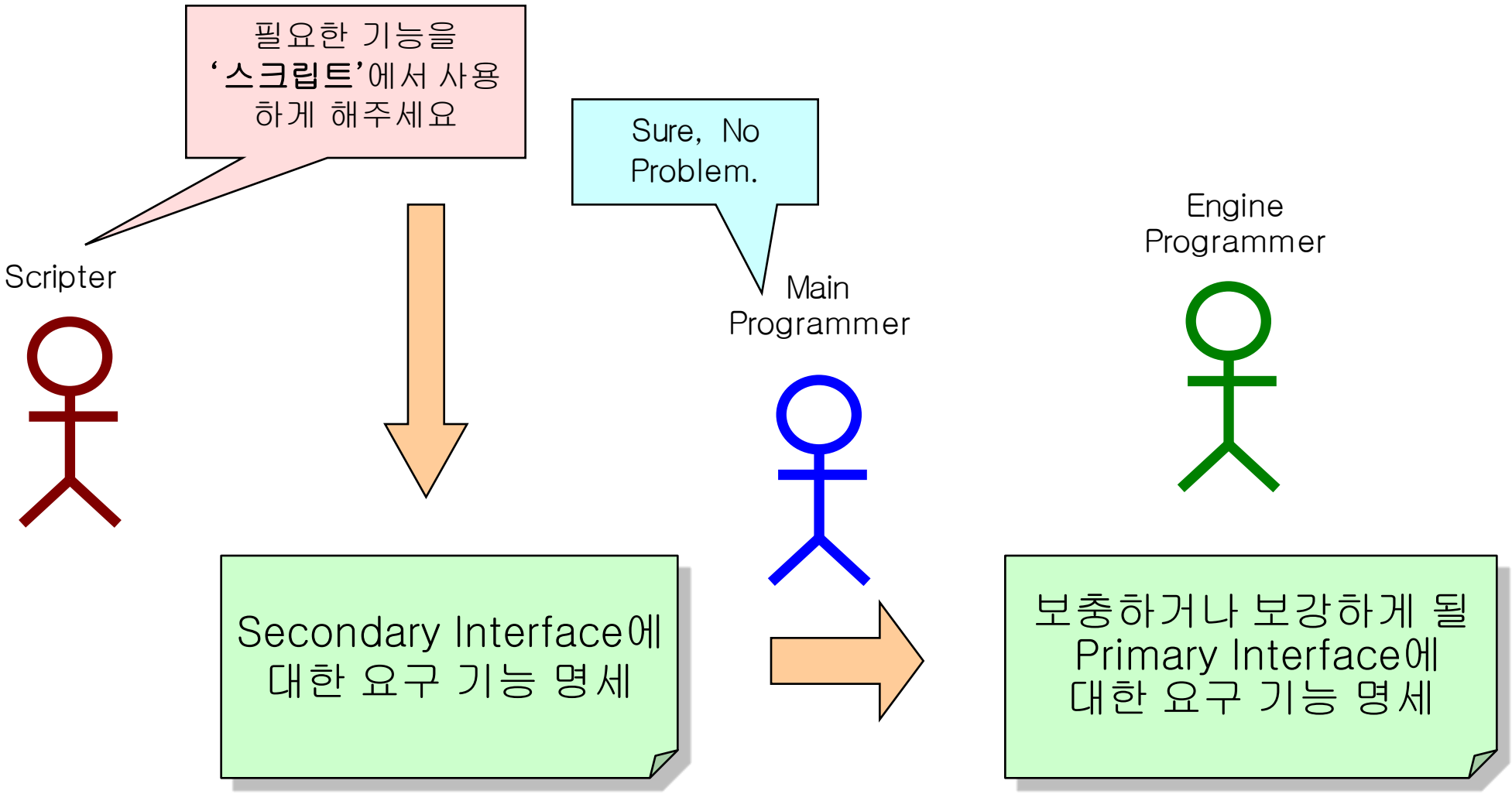


GaLic. Heesung Oh
009, Copyright All rights reserved

- Main Programmer가 인터페이스를 미리 작성해서 스크립터에게 전달
- 스크립터는 스크립트를 작성. 필요한 내용에 대한 추가요구
- ※ 인터페이스는 스크립터, 메인 프로그래머의 독립된 작업을 돕는다.



GaLic. Heesung Oh
2009, Copyright All rights reserved



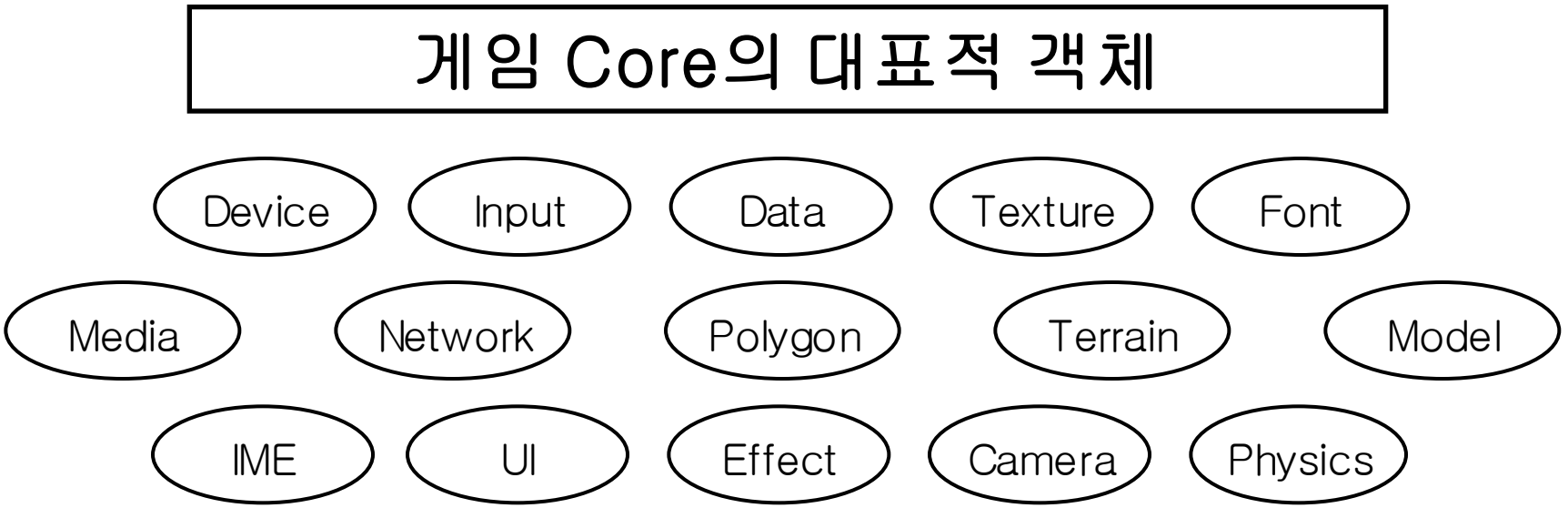
2009, Copyright All rights reserved

PART 3.

게임 엔진 제작

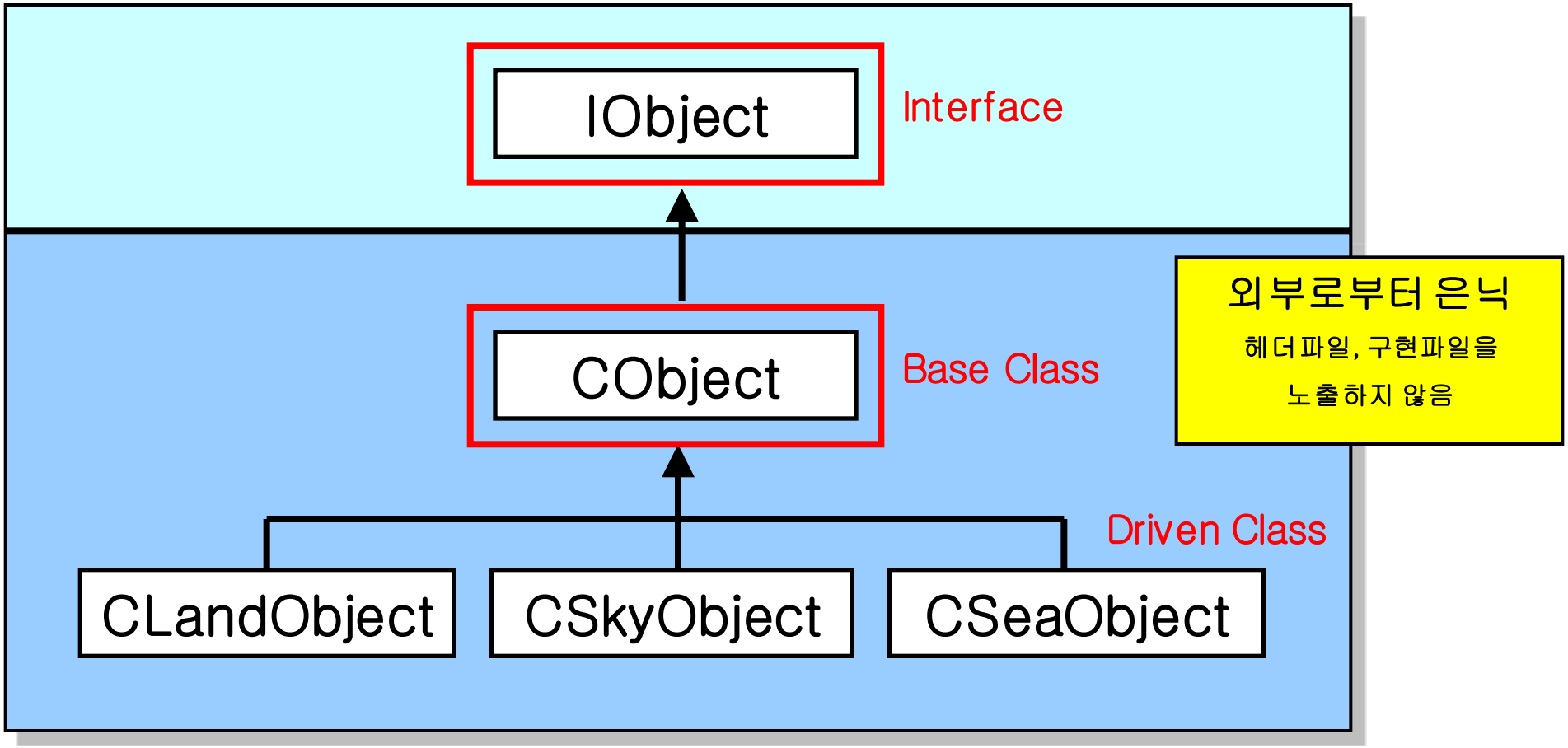
GaLic. Heesung Oh
2009, Copyright All rights reserved

- C++과 Interface
  - 제어와 감시를 목적으로 설계된 추상 클래스
  - 다른 응용 프로그램과 통신을 위한 접합 점



GaLic. Heesung Oh
2009, Copyright All rights reserved

- Interface객체
  - 순수 가상 함수로 구성된 Interface 객체는 최상위 클래스로 항상 유지



GaLic. Heesung Oh
2009, Copyright All rights reserved

- 추상 Interface Class
  - Interface는 클래스내부의 데이터 생성과 초기화를 위한 Create,
  - Create에서 생성한 데이터를 해제하는
  - Destroy 한 쌍으로 구성
  - 하위클래스에서 사용되는
  - 공통 메소드를 가상함수로 작성
  - 멤버변수를 두지 않음

Example C++ Code

```
typedef struct interface;

interface IObject
{
        ...
        virtual Create() = 0;
        virtual Destroy() = 0;

        ...
        virtual FrameMove()= 0;
        virtual Render() = 0;
        ...
};
```

GaLic. Heesung Oh
2009, Copyright All rights reserved

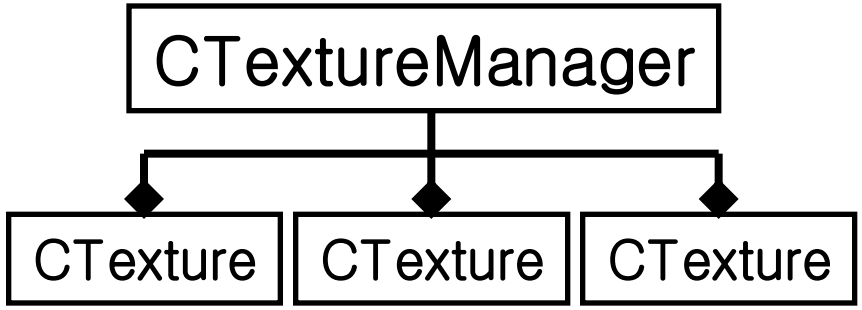
- Interface Class
  - Interface의 순수 가상함수와 ‘Query’ 함수
  - Interface는 객체의 특징을 나타내거나 공용적인 메소드만 순수 가상함수로 작성
  - 그 이외의 메소드는 Query함수를 만들어 질의어 형식으로 사용

“Query” Example C++ Code

```
Query( char* card, void* pData );
{
    if( 0 == stricmp( "RunLoadFunc", Card ) )
    {
        LoadFunc();
    }
}
```

GaLic. Heesung Oh
2009, Copyright All rights reserved

- Manager Class
  - 다수의 객체를 요구하는 경우 Manager로 관리
- 객체의 생성 메커니즘
  - Manager는 관리 객체를 생성하는 메소드 제공
- 객체의 Find 알고리즘
  - Manager는 관리 객체를 검색해서 반환(return) 하는 메소드 제공
  -



CManager::Generate()

CManager::Get(INT index)

CManager::Get(LPSTR szKey)

CManager::Get(Enum eType)

GaLic. Heesung Oh
2009, Copyright All rights reserved

Manager 초기화

메니저를 생성하는 메커니즘을 제공해야 한다.

Manager Interface의 경우 설정 파일을 만들어, 구동 시 파일을 읽어서 내용을 초기화 하는 경우도 있다.

File : System/Manager.ini

```
Name     : CameraManager
Type     : Third
Position_X        : 0
Position_Y        : 50
Position_Z        : -50
Target   : Player
```

Example C++ Code

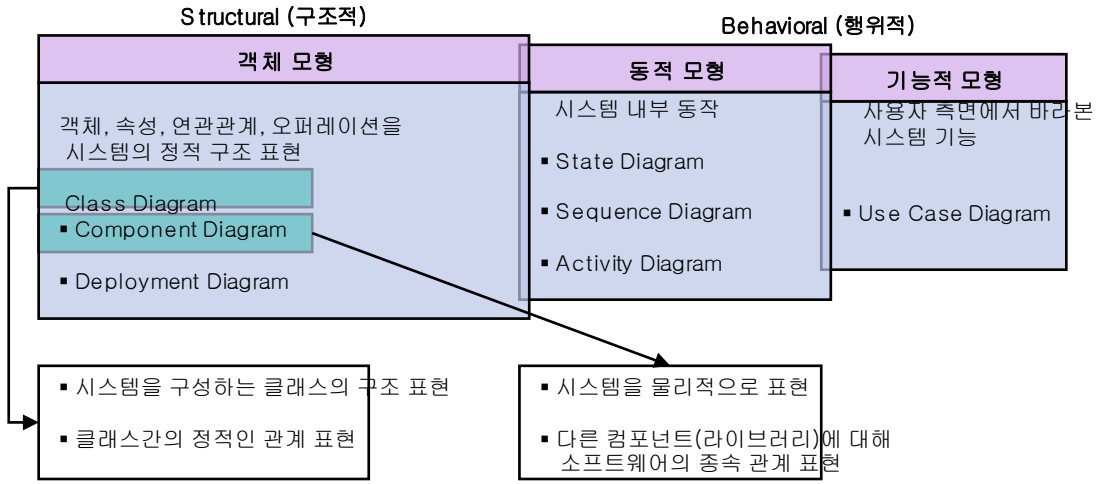
```
CreateManager( “CameraManager”, “System/Manager.ini”, &pData );
```


GaLic. Heesung Oh
2009, Copyright All rights reserved

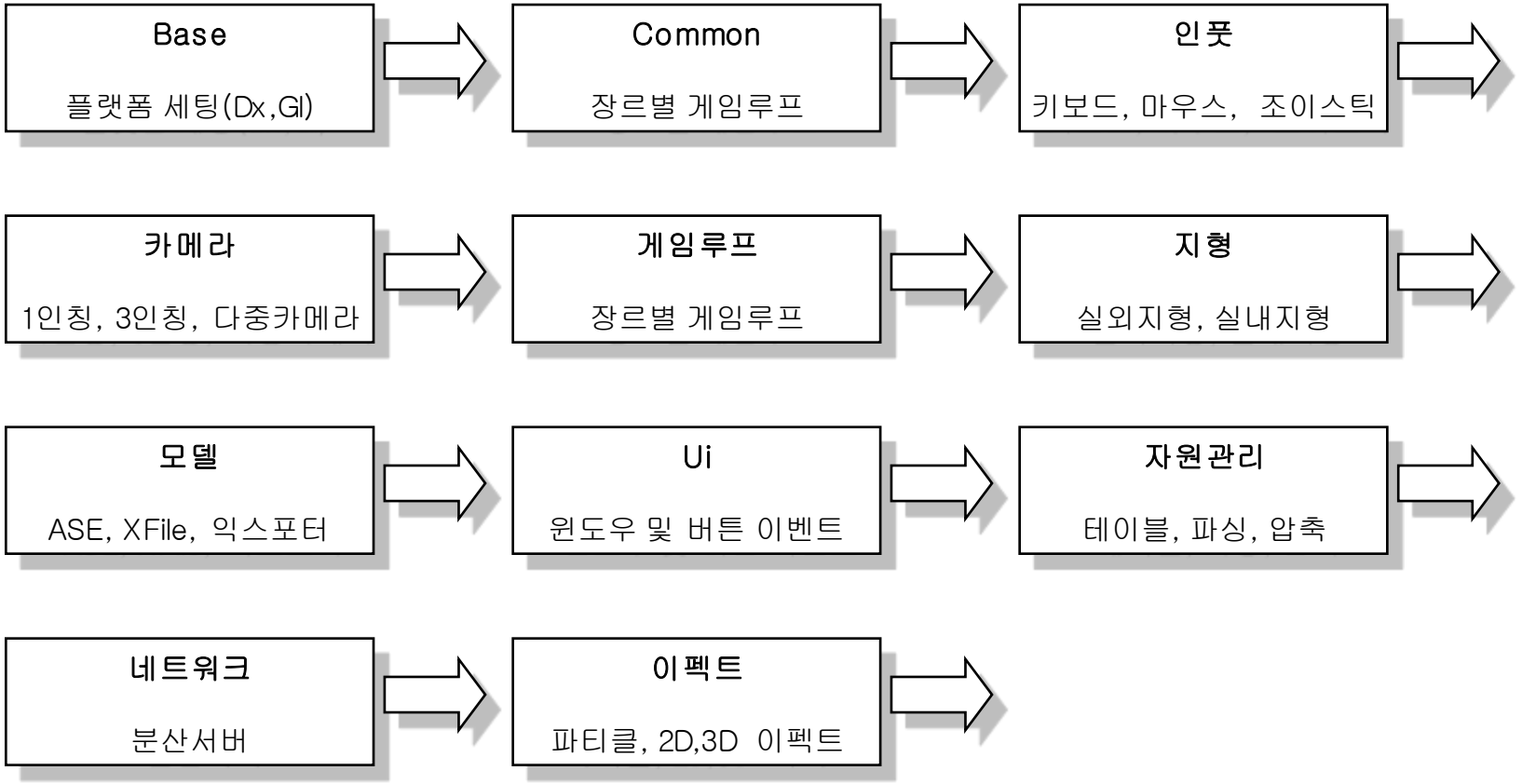
- 자료구조 정리
  - 자료구조의 중복을 해소
  - 프로그램의 통일성 보장

- UML활용
  - UML은 만능이 아님
  - 하지만 있으면 개별작업, 팀 작업에 유리
  - 핵심 함수와 변수만 기입

- UML 예제



GaLic. Heesung Oh
2009, Copyright All rights reserved



- 게임 제작과 병행하여 순환 식으로 계속 모듈들을 갱신

2009, Copyright All rights reserved

PART 4.

엔진 제작 연습

GaLic. Heesung Oh
2009, Copyright All rights reserved

- 라이브러리란
  - 컴파일 된 소스코드
  - 엔진을 구성하는 모듈
- 목적
  - 모듈의 응집성,독립성 향상
  - 개발의 효율성
  - 코드의 재활용과 확장성
- 2D 게임 라이브러리
  - 초보자에게 유리
  - 3D로 확장하기 쉬움
- 라이브러리 예제

GaLic. Heesung Oh
2009, Copyright All rights reserved

- 동적 결합과 인터페이스를 이용한 모델 예제
  - 2D 모델
  - 3D 모델1
  - MD2
  - MD3

- 기타 응용사례
  - 다중 플랫폼: OpenGL, DirectX, Xbox, 기타…
  - 카메라: 1인칭, 3인칭
  - 모델: …
  - 이펙트
  - 게임데이터 등등

GaLic. Heesung Oh
2009, Copyright All rights reserved

- 계층적 자료구조인 트리 자료구조를 렌더링에서 자주 사용

- "장면관리자는 트리다!!!"
  - 2개의 Dinked 리스트 이용

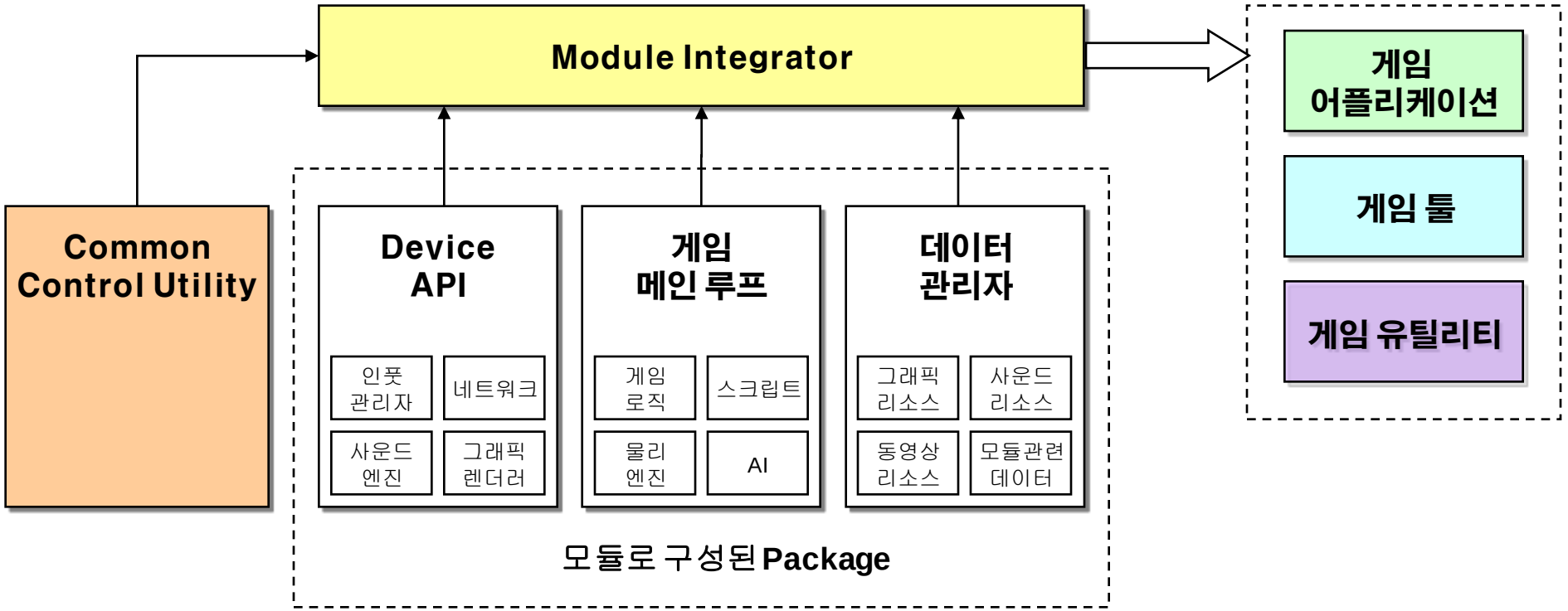
- 응용 사례
  - User Interface
  - 모델 애니메이션
  - 장면관리자
  - 컬링, 렌더링
  - 기타 두루 쓰임

트리의 구조

```
class CLnTree // Ln tree class
{
public:
 CLnTree* pP; // 부모노드
 CLnTree* pC; // 자식노드
 CLnTree* pB; // 자매노드 이전
 CLnTree* pN; // 자매노드 다음

public:
 CLnTree() : pP(0), pC(0), pB(0), pN(0){}
 virtual ~CLnTree();

 bool HasParent() { return (0 != pP); }// 부모가 있냐?
 bool HasNotParent() { return (0 == pP); }
 bool HasChild() { return (0 != pC); } // 자식이 있냐?
 bool HasNotChild() { return (0 == pC); }
 bool IsSiblingF() { return (pP && pP->pC == this); }// 내가 첫번째 아들이냐

 bool IsSiblingL() { return (pP && 0 == pN) ; } // 내가 마직막 아들이냐
 bool IsRoot() { return (0 == pP); }// 내가 루트인가?
 bool IsChild() { return (0 != pP); }// 내가 자식인가?
 CLnTree* GetSiblingF() { return (pB)? pB->GetSiblingF(): this; }// 자매의 처음 포인터를 찾는다.
 CLnTree* GetSiblingL() { return (pN)? pN->GetSiblingL(): this; }// 자매의 마지막 포인터를 찾는다
 CLnTree* FindRoot() { return (pP)? pP->FindRoot(): this; }// 최상위 루트

 void Attach(CLnTree *pCh);
 void Detach();
 int CountNodes();
};
```

GaLic. Heesung Oh
2009, Copyright All rights reserved

- 엔진에 기반한 장르별 게임코드 생성



2009, Copyright All rights reserved