Getting Started OpenGL

OpenGL이 무엇인가?

-API(application programming interface)로 간주, 실제로는 API가 아니라 Khronos Group에서 개발, 관리 하는 사양임

-각 함수의 결과/출력이 어떻게 수행되어야 하는지 정확하게 특정해 놓음

-버그가 발생하면 대부분 그래픽카드 드라이버 업데이트를 통해 해결 (최신 버전의 OpenGL이 포함되어짐)

Core-profile vs Immediate mode

-옛날에는 OpenGL을 통해 개발하는 것은 Immediate Mode(가끔 고정 함수 파이프라인)를 의미함

-Immediate Mode : 그래픽을 그리는데 사용하기 쉬운 방법, 비효율적, 3.2부터 사용하지 않게 됨

-OpenGL 버전3.2 부터 Core-profile를 사용하게 됨

-Core-profile : 사용하지 않는 이전 기능을 제거한 OpenGl 사양의 한 부분

Extensions 확장

-그래픽 회사에서 새로운 기술이나 큰 최적화 렌더링을 낼 때마다 이것들은 드라이버의 Extension 기능에서 구현이 됨

-그래픽 개발자는 Extensions가 지원이 되는지 확인하는 것으로 렌더링 기술을 사용할 수 있음

|  |
| --- |
| if(GL\_ARB\_extention\_name)  {  // 하드웨어가 지원하는 새로운 기술, 최적화 렌더링  }  else  {  // 지원하지 않는 Extension, 오래된 방법으로 사용됨  } |

State Machine

-OpenGL은 자체로 하나의 큰 스테이트 머신으로, 작동 방식을 정의하는 변수의 모음임

-일부 옵션을 설정, 일부 버퍼를 조작한 뒤, 현재 컨택스트를 사용해 상태를 변경하는 경우가 많음

ex) 삼각형 대신 선을 그리고 싶다 > 변수를 변경, OpenGL상태를 변경 > OpenGL에 선을 그려야 한다고 컨텍스트를 변경 > 다음 그리기 명령: 삼각형대신 선을 그림

Object 오브젝트

-OpenGL은 C로 작성되어 짐

-오브젝트들, 객체들은 C와 같은 구조체의 모습으로 시각화 할 수 있음

|  |
| --- |
| struct object\_name{  float option1;  int option2;  char[] name;  } |

-사용 예시

|  |
| --- |
| // OpenGL의 스테이트  struct OpenGL\_Context {  ...  object\_name\* object\_Window\_Target;  ...  }; |
| // 오브젝트 객체 생성, id로 저장  unsigned int objectId = 0;  glGenObject(1, &objectId);  // id를 이용해서 객체를 context에 바인딩  glBindObject(GL\_WINDOW\_TARGET, objectId);  // GL\_WINDOW\_TARGET을 이용해 창 옵션을 변경  glSetObjectOption(GL\_WINDOW\_TARGET, GL\_OPTION\_WINDOW\_WIDTH, 800);  glSetObjectOption(GL\_WINDOW\_TARGET, GL\_OPTION\_WINDOW\_HEIGHT, 600);  // ID를 0으로 설정, 객체 바인딩 해제  glBindObject(GL\_WINDOW\_TARGET, 0); |