Lighting-Colors

-색은 RGB( Red, Green, Blue )로 나타내어짐

|  |
| --- |
| glm::vec3 coral(1.0f, 0.5f, 0.31f); |

-현실 세계에서의 물질의 색

-만약에 빨간색의 물질이라면 태양광을 비췄을 때, 빨강을 제외한 나머지 색을 흡수, 빨강색만을 반사하여 그것을 보는 사람은 빨강색이라고 인식함

-이러한 반사 규칙은 그래픽 세계에서도 적용됨

-광원에 컬러를 설정할 수 있음

-오브젝트에 컬러를 설정할 수 있음

-실제로 나오는 색은 (광원의 컬러) \* (오브젝트 컬러)

|  |
| --- |
| glm::vec3 lightColor(1.0f, 1.0f, 1.0f); //흰색 광원  glm::vec3 toyColor(1.0f, 0.5f, 0.31f);  glm::vec3 result = lightColor \* toyColor; // = (1.0f, 0.5f, 0.31f); |
| glm::vec3 lightColor(0.0f, 1.0f, 0.0f); //녹색 광원  glm::vec3 toyColor(1.0f, 0.5f, 0.31f);  glm::vec3 result = lightColor \* toyColor; // = (0.0f, 0.5f, 0.0f); |

A lighting scene

-이전에 공부한 큐브생성은 생략

-큐브는 1개만 생성, 텍스처 제거

-광원큐브를 만들기 위한 새로운VAO

|  |
| --- |
| unsigned int lightVAO;  glGenVertexArrays(1, &lightVAO);  glBindVertexArray(lightVAO);  glBindBuffer(GL\_ARRAY\_BUFFER, VBO);  glVertexAttribPointer(0, 3, GL\_FLOAT, GL\_FALSE, 3 \* sizeof(float), (void\*)0);  glEnableVertexAttribArray(0); |

-fragment shader source

|  |
| --- |
| #version 460 core  out vec4 FragColor;    uniform vec3 objectColor;  uniform vec3 lightColor;  void main()  {  FragColor = vec4(lightColor \* objectColor, 1.0);  } |

-광원, 오브젝트 컬러는 uniform으로 받음

-shader의 uniform에 데이터 전달

|  |
| --- |
| lightingShader.use();  lightingShader.setVec3("objectColor", 1.0f, 0.5f, 0.31f);  lightingShader.setVec3("lightColor", 1.0f, 1.0f, 1.0f); |

-광원큐브, 램프는 값이 변하지 않기를 원함

-shader을 하나 더 추가

|  |
| --- |
| #version 330 core  out vec4 FragColor;  void main()  {  FragColor = vec4(1.0); // 4 개의 모든 벡터의 값을 1.0으로 설정합니다.  } |

-광원의 위치

|  |
| --- |
| glm::vec3 lightPos(1.2f, 1.0f, 2.0f); |

-광원의 model

|  |
| --- |
| model = glm::mat4();  model = glm::translate(model, lightPos);  model = glm::scale(model, glm::vec3(0.2f)); |

-랜더링 코드

|  |
| --- |
| lampShader.use();  // model, view, projection 행렬 unifrom을 설정하세요.  ...  // 램프 오브젝트를 그립니다.  glBindVertexArray(lightVAO);  glDrawArrays(GL\_TRIANGLES, 0, 36); |

