Программирование шейдеров для движка axelynx.

В движке есть предустановленная система стандартных шейдеров, но зачастую разработчику нужно более гибкое поведение программы, чем набор предустановленных шейдеров, в этом случае разработчикам нужно писать свои шейдеры. За этим словом стоит относительно простая техника создания выполняемых на видеокарточке программ. Важно понимать, что шейдер в терминологии axelynx это некое предустановленное состояние конвейера, т. е. Слинкованные вместе вершинные и фрагментные шейдеры как минимум, ну и геометрический или тесселяционный как максимум. По сути эти мини-программки просто принимают какие-то данные на входе, обрабатывают их, и посылают дальше. Например принимают локальные координаты вершины, перемножют их на mvp матрицу и передают фрагментному шейдеру. Фрагментный шейдер данные, полученные от вершины линейно интерполирует от вершины к вершине. Есть три типа данных для шейдера. in входящие данные, например:

```
in vec3 somevalue;
```

шейдер принимает вектор от предыдущего шейдера, состоящий из трех флоатов. Например позицию вершины, или еще что либо.

```
out vec4 frag_color;
```

шейдер дает следующему шейдеру на выход вектор, состоящий из четырех флоатов, например цвет фрагмента. Напоминаю, что если это выход с фрагментного шейдера, то значение будет интерполироваться.

```
uniform mat4 viewmatrix;
```

шейдер принимает переменную от движка. Например матрицу вида.

Для фрагментного шейдера (последнего в конвейере) модификатором out обозначаются выходные данные конечного цвета. Для вершинного шедера (первого в конвейере) входные данные — это аттрибуты вершин.

Есть набор зарезервированных движком имен для юниформов. Они посылаются движком, устанавливать их вручную нет смысла.

```
mat4 model
матрица модели. Сюда пишется глобальная матрица трансформации какой-либо entity. Чтобы получить мировые координаты в пространстве сцены надо умножить позицию вершины на эту матрицу.

mat4 modelview
матрица трансформации вида. Сюда пишется трансформация вместе с камерой. Чтобы получить позицию вершины в пространстве камеры, надо ее множить на эту матрицу.

mat4 modelviewproj
матрица трансформации. Переводит позицию вершины в ортонормированный куб (грубо говоря в куб с позициями -1,-1,-1,1,1,1. Дальше фрагменты выводятся в соответствии с вьюпортом.
```

```
mat3 normalmatrix
матрица, хранящая поворот объекта и его скейл.

vec3 entitycolor
локальный цвет объекта. Четвертым компонентом идет альфа.

vec3 viewdir
направление камеры в координатах сцены

vec3 lightpos
позиция света в координатах сцены. Движокм не передается

vec3 lightdir
направление света в координатах сцены. Движком не передается

vec3 eyepos
позиция камеры в координатах сцены.
```

Аттрибуты вершины, такие как позиция, нормаль, текстурные координаты должны явно объявляться в шейдере. Движок объявляет несколько дефайнов с системными номерами вершин. Как то так:

```
vec3 layout(location=VA_POSITION)

vec3 layout(location=VA_NORMAL)

vec3 layout(location=VA_TANGENT)

vec3 layout(location=VA_COLOR)

vec3 layout(location=VA_TEXCOORD0)

vec3 layout(location=VA_TEXCOORD1)

vec3 layout(location=VA_BONEID)

vec3 layout(location=VA_WEIGHT)

vec3 layout(location=VA_POINT_SIZE)

vec3 layout(location=VA_NEXT_POSITION)

vec3 layout(location=VA_NEXT_NORMAL)

vec3 layout(location=VA_NEXT_TANGENT)

vec3 layout(location=VA_NEXT_TEXCOORD0)
```