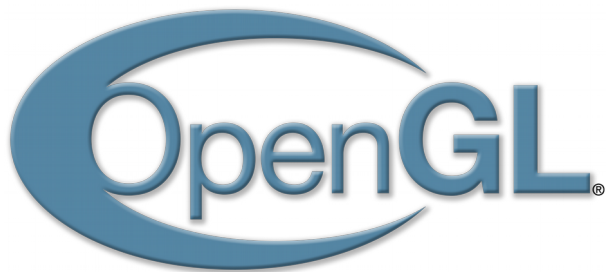


OpenGL

Режими



Режими на работа в OpenGL

- **Render** – Визуализация (по подразбиране);
- **Select** – Индентификация на елементи;
- **Feedback** – Получаване на примитиви.

Избор на режим

```
GLint glRenderMode (GLenum mode)
```

mode :

GL_RENDER

Визуализация

GL_SELECT

Идентификация

GL_FEEDBACK

Примитиви

връща : брой записи от пред. режим

при брой < 0, значи буфера е бил препълнен

Схема на работа RENDER

`glRenderMode (GL_RENDER) ;`

Режим по подразбиране.

Командите се изпълняват и резултата от тях променя вътрешното състояние на OpenGL, включително и Буфера на цвета.

Схема на работа SELECT

1. Визуализация в режим `GL_RENDER`;
2. `glSelectBuffer(...)`;
3. `glRenderMode(GL_SELECT)`;
4. Визуализация в режим `GL_SELECT`
(използват се команди за задаване на имена на визуализираните обекти);
5. `Брой = glRenderMode(GL_RENDER)`;
6. Обработка на `SelectBuffer`-а.

Именуване

```
void glInitNames (void)
```

```
void glPushName (GLuint name)
```

```
void glPopName (void)
```

```
void glLoadName (GLuint name)
```

Стек с имената

- Използва се за именуване на обектите в йерархични сцени;
- Стека с имената съдържа цели числа;
- Цялото моментно състояние на стека идентификация на визуализираните в момента примитиви;
- До промяната на стека всички обекти са с едно и съща идентификация, независимо колко сложни са те и от колко примитиви се състоят.

glInitNames()

N_4
N_3
N_2
N_1



Празен стек
с имена

glPushName(N)

N_4
N_3
N_2
N_1



N
N_4
N_3
N_2
N_1

glPopName()

N_4
N_3
N_2
N_1



N_3
N_2
N_1

glLoadName(N)

N_4
N_3
N_2
N_1



N
N_3
N_2
N_1

Пример

```
glInitNames();  
glPushName(0);
```

```
glLoadName(1);  
gluSolidSphere(1, 32, 32);
```

```
glLoadName(2);  
gluSolidSphere(2, 32, 32);
```

```
glLoadName(3);  
glPushName(1);  
glBegin(GL_TRIANGLES);
```

```
    glVertex3d(0,0,0);
```

```
    glVertex3d(1,0,0);
```

```
    glVertex3d(1,1,0);
```

```
glEnd();
```

```
glLoadName(2);  
gluSolidSphere(1, 32, 32);  
glPopName();
```

1

2

3 1

3 2

Схема на работа SELECT

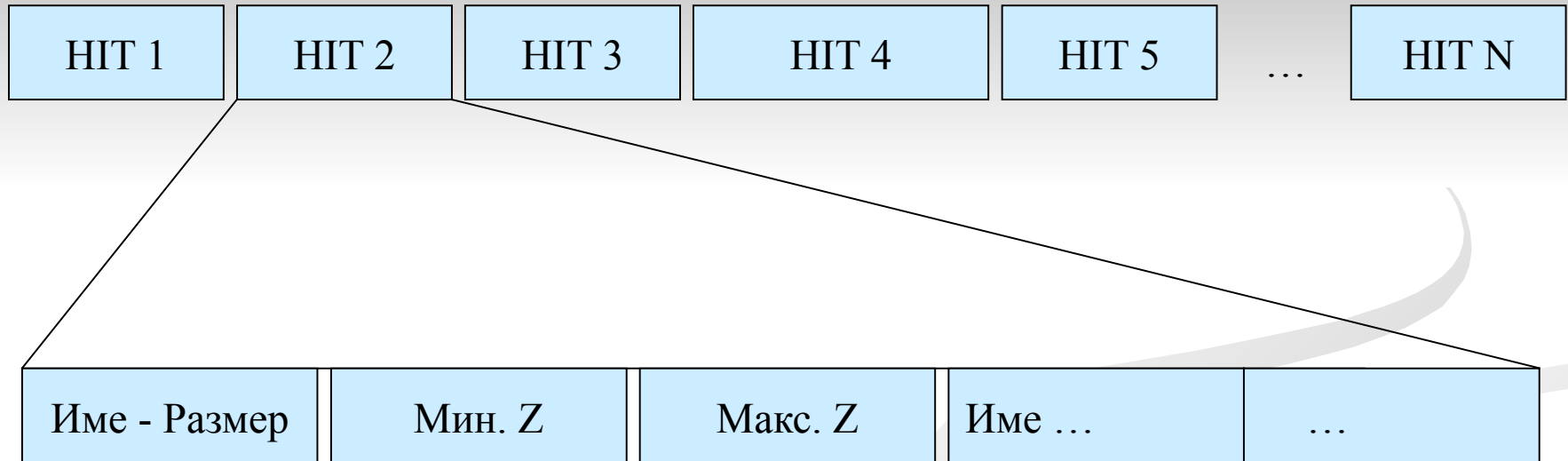
```
void glSelectBuffer(  
    GLsizei size, GLuint *buffer);
```

size: размер на буфера
buffer: буфер

Структура на SelectBuffer

- Буфера се състои от N на брой попадения (Hits) – колко са ни връща `glRenderMode(...)`, когато излизаме от SELECT режима;
- Всеки Hit описва един елемент на изображението (с едно име) – размер на името, минимална и максимална Z стойност, състояние на стека с имената;
- Всеки полета в Hit записа са от тип `GLuint`.

Структура на SelectBuffer



Мин. Z = Мин. екранен Z * (2³²-1)

Броя на елементите в името се определят от първото поле на Hit записа

Пример (1/3)

```
// Визуализация с включено именуване на елементите
void draw() {
    glLoadIdentity();
    glInitNames();
    glPushName(1);
    glutSolidSphere(1, 32, 32);
    glLoadName(2);
    glTranslated(2,0,0);
    glutSolidSphere(1, 32, 32);
}

void display() {
    glClearColor(0,0,0,0);
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    draw();
    glutSwapBuffers();
}
```


Пример (2/3)

```
#define BUFSIZE 1000

void mouse(int button, int state, int x, int y) {
    GLuint selectBuffer[BUFSIZE]; GLint hits; GLint vp[4];

    glGetIntegerv(GL_VIEWPORT, vp);
    glSelectBuffer(BUFSIZE * sizeof(GLuint), selectBuffer);
    glRenderMode(GL_SELECT);
    glMatrixMode(GL_PROJECTION);
    glPushMatrix();
    glLoadIdentity();
    gluPickMatrix((GLdouble)x, (GLdouble)(vp[3] - y), 5.0, 5.0, vp);
    // тут используй проекция от resize ( glOrtho, gluLookAt, ...)
    glMatrixMode(GL_MODELVIEW);
    draw();
    glMatrixMode(GL_PROJECTION);
    glPopMatrix();
    glMatrixMode(GL_MODELVIEW);
    glFlush();
    hits = glRenderMode(GL_RENDER);
    processHits(hits, selectBuffer);
}
```

Пример (3/3)

```
void processHits(GLint hits, GLuint buffer[]) {
    int i; unsigned int j; GLuint names, *ptr;

    printf ("hits = %d\n", hits);
    ptr = (GLuint *) buffer;
    for (i = 0; i < hits; i++) {
        names = *ptr;
        printf (" name size for hit = %d\n", names); ptr++;
        printf(" min-z = %g;", (float) *ptr/0x7fffffff); ptr++;
        printf(" max-z = %g\n", (float) *ptr/0x7fffffff); ptr++;
        printf (" name: ");
        for (j = 0; j < names; j++) {
            printf ("%d ", *ptr);
            ptr++;
        }
        printf ("\n");
    }
}

// В main перед glutMainLoop добавьте
glutMouseFunc(mouse);
```

gluPickMatrix(x, y, w, h, vp)

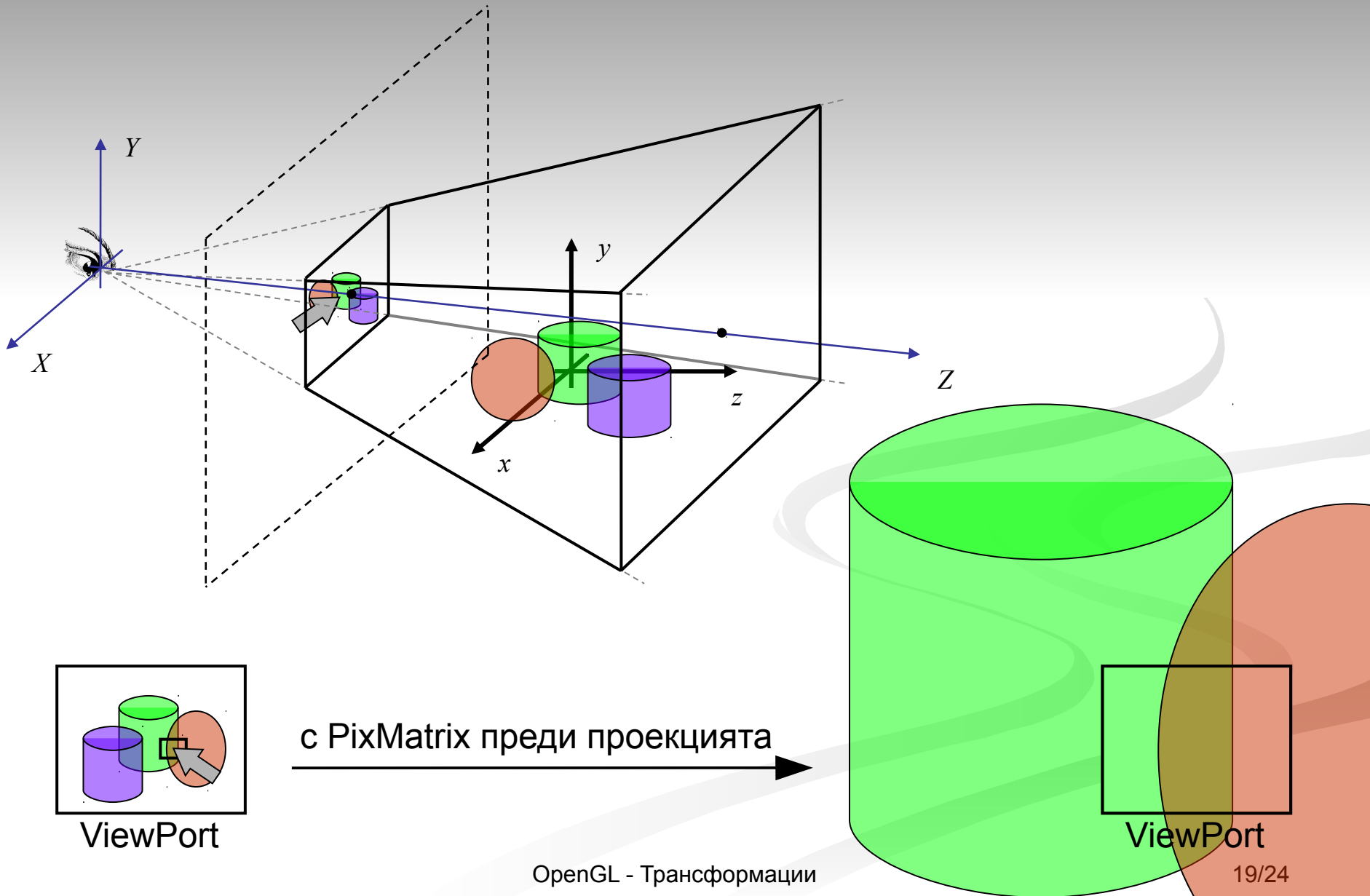


Схема на работа FEEDBACK

```
void glFeedbackBuffer(GLsizei size, GLenum type,  
                     GLfloat *buffer)
```

size: размер на буфера

type:

GL_2D

x, y

GL_3D

x, y, z

GL_3D_COLOR

x, y, z, r, g, b, a

GL_3D_COLOR_TEXTURE

x, y, z, r, g, b, a, t_{1...4}

GL_4D_COLOR_TEXTURE

x, y, z, w, r, g, b, a, t_{1...4}

buffer: буфер

Примитиви в буфера

Тип	Код	Данни
Точка	GL_POINT_TOKEN	Vertex
Линия	GL_LINE_TOKEN GL_LINE_RESET_TOKEN	Vertex, Vertex
Многоъгълник	GL_POLYGON_TOKEN	N, Vertex, ...
Битмап	GL_BITMAP_TOKEN	Vertex
Пиксел Правоъгълник	GL_DRAW_PIXEL_TOKEN GL_COPY_PIXEL_TOKEN	Vertex
Pass-through	GL_PASS_THROUGH_TOKEN	GLfloat число

Структура на FeedbackBuffer

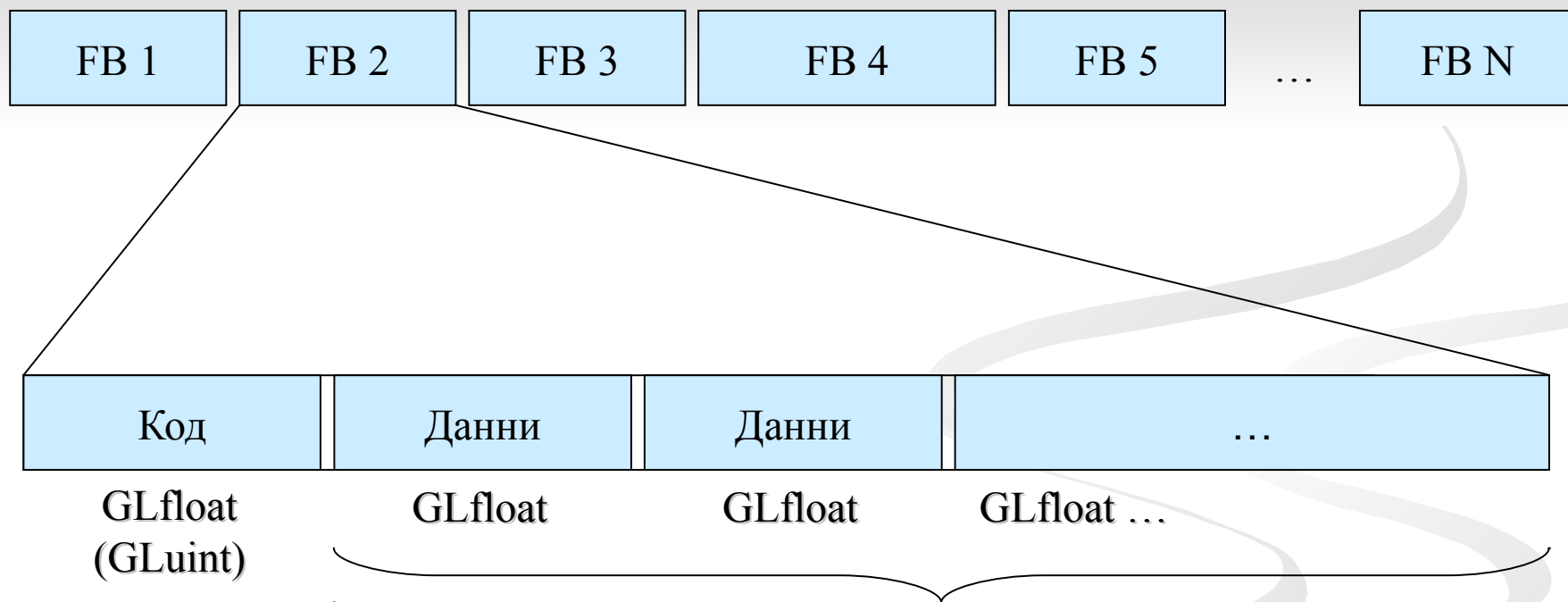


Схема на работа FEEDBACK

```
void glPassThrough(GLfloat token)
```

Записва в feedback буфера
`GL_PASS_THROUGH_TOKEN` с
параметър стойността на `token`.

OpenGL - Режими

Въпроси?