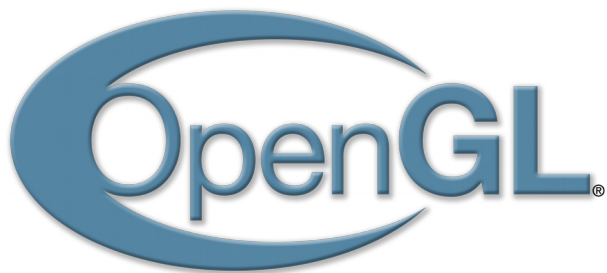




# OpenGL

Изграждане на  
Геометрични обекти 2



# Анимация (1/3)

`glutTimerFunc`

`glutIdleFunc`

`glutPostRedisplay`

`glutPostWindowsRedisplay`

...

+

**Промяна на визуализацията в `display`**

# Анимация (2/3)

```
int angle = 0;

void display(void)
{
    glClearColor(0,0,0,1);
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    glLoadIdentity();
    glRotated(angle, 0,1,0);

    glColor3d(1,1,1);
    glutSolidTeapot(1);

    glutSwapBuffers();
}

void timer(int value)
{
    angle++;
    glutPostRedisplay();
    glutTimerFunc(50, timer, 1);
}

// В main ...
glutTimerFunc(50, timer, 1);
```

# Анимация (3/2)

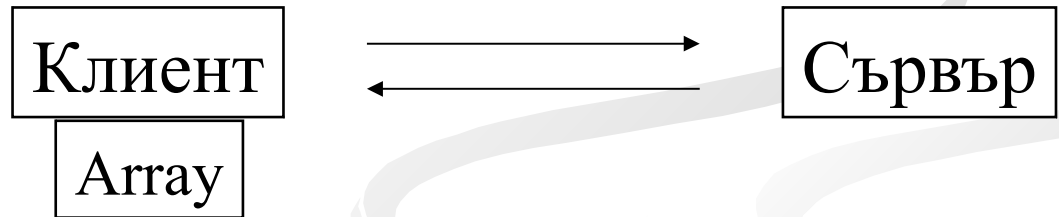


# Масиви и буфери с данни

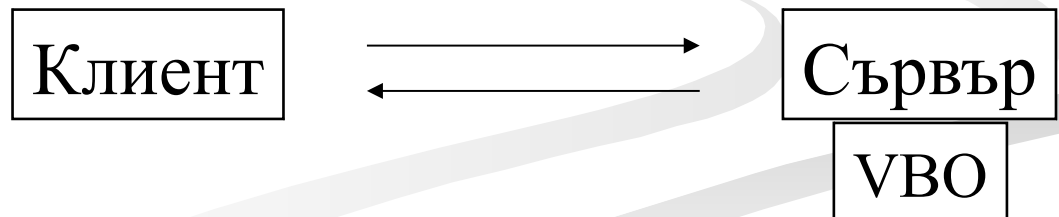
## 1. Команди



## 2. Масиви в клиента



## 3. Буфери/обекти в сървъра



# Задаване на масиви с данни

```
void glEnableClientState(GLenum caps)  
void glDisableClientState(GLenum caps)
```

caps:

GL\_VERTEX\_ARRAY

GL\_COLOR\_ARRAY

GL\_NORMAL\_ARRAY

GL\_TEXTURE\_COORD\_ARRAY

GL\_EDGE\_FLAG\_ARRAY

GL\_INDEX\_ARRAY

GL\_FOG\_COORD\_ARRAY

GL\_SECONDARY\_COLOR\_ARRAY

# Клиентско състояние (1/3)

```
void glVertexPointer(  
    GLint size,  
    GLenum type,  
    GLsizei stride,  
    const GLvoid *pointer)
```

```
void glGetPointerv(GLenum pname,  
    GLvoid* *params)
```

# Клиентско състояние (2/3)

`glColorPointer`

`glIndexPointer`

`glNormalPointer`

`glTexCoordPointer`

`glEdgeFlagPointer`

без размер и тип

```
void glGetPointerv(GLenum pname,  
GLvoid* *params)
```



# Клиентско състояние (3/3)

**size:**

**1,2,3,4** – брой координати

**type:**

**GL\_SHORT, GL\_INT, GL\_DOUBLE, ...**

**stride:**

**0 или повече** – отместване между два ел.



# Позоваване на елемент

```
void glVertexElement(GLint i)
```

# Пример (1/2)

```
static GLdouble vertices[] = {  
    // XYZH  
    -1.0, -1.0, -1.0, 1.0,  
    1.0, -1.0, -1.0, 1.0,  
    0.0, 1.0, -1.0, 1.0,  
    0.0, 0.0, 1.0, 1.0  
};  
static GLfloat colors[] = {  
    // RGB  
    1.0, 0.0, 0.0,  
    0.0, 1.0, 0.0,  
    1.0, 1.0, 1.0,  
    0.0, 0.0, 1.0  
};
```

# Пример (2/2)

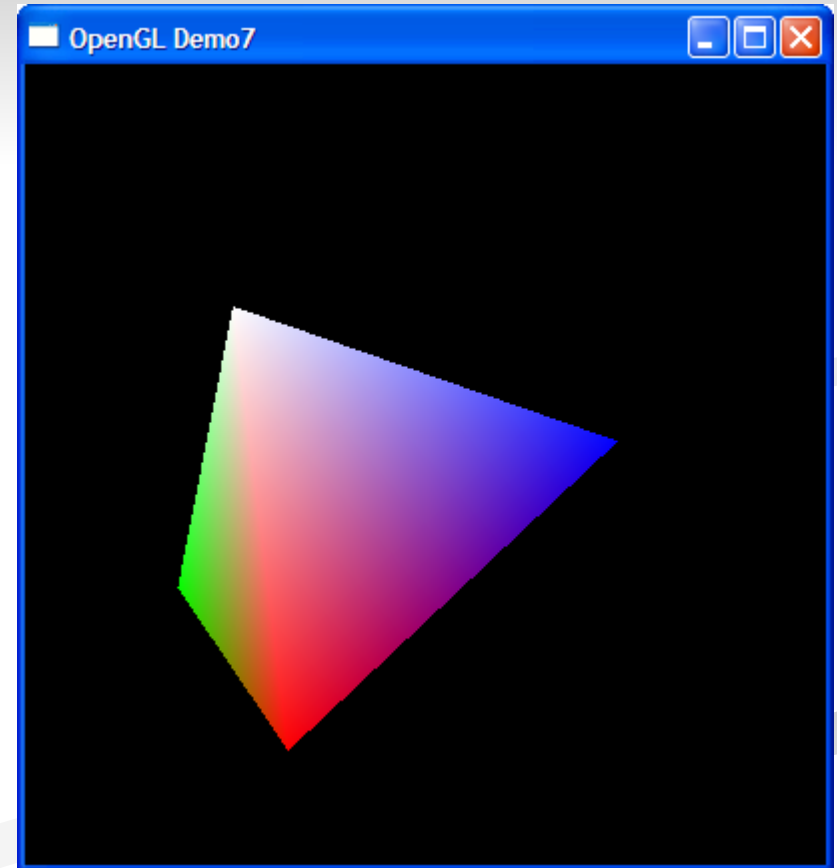
```
glEnableClientState(GL_VERTEX_ARRAY);
glEnableClientState(GL_COLOR_ARRAY);
glVertexPointer(4, GL_DOUBLE, 0, vertices);
glColorPointer(3, GL_FLOAT, 0, colors);

glBegin(GL_TRIANGLES);
    glVertexElement(0);
    glVertexElement(1);
    glVertexElement(2);

    glColorElement(0);
    glColorElement(1);
    glColorElement(3);

    glColorElement(1);
    glColorElement(2);
    glColorElement(3);

    glColorElement(2);
    glColorElement(0);
    glColorElement(3);
glEnd();
```



# Пример (1/2)

```
struct {
    GLdouble vertex[4];
    GLfloat color[3];
} points[] = {
    // XYZH, RGB
    {{-1.0, -1.0, -1.0, 1.0}, {1.0, 0.0, 0.0}},
    {{1.0, -1.0, -1.0, 1.0}, {0.0, 1.0, 0.0}},
    {{0.0, 1.0, -1.0, 1.0}, {0.0, 0.0, 1.0}},
    {{0.0, 0.0, 1.0, 1.0}, {1.0, 1.0, 1.0}}
};
```

# Пример (2/2)

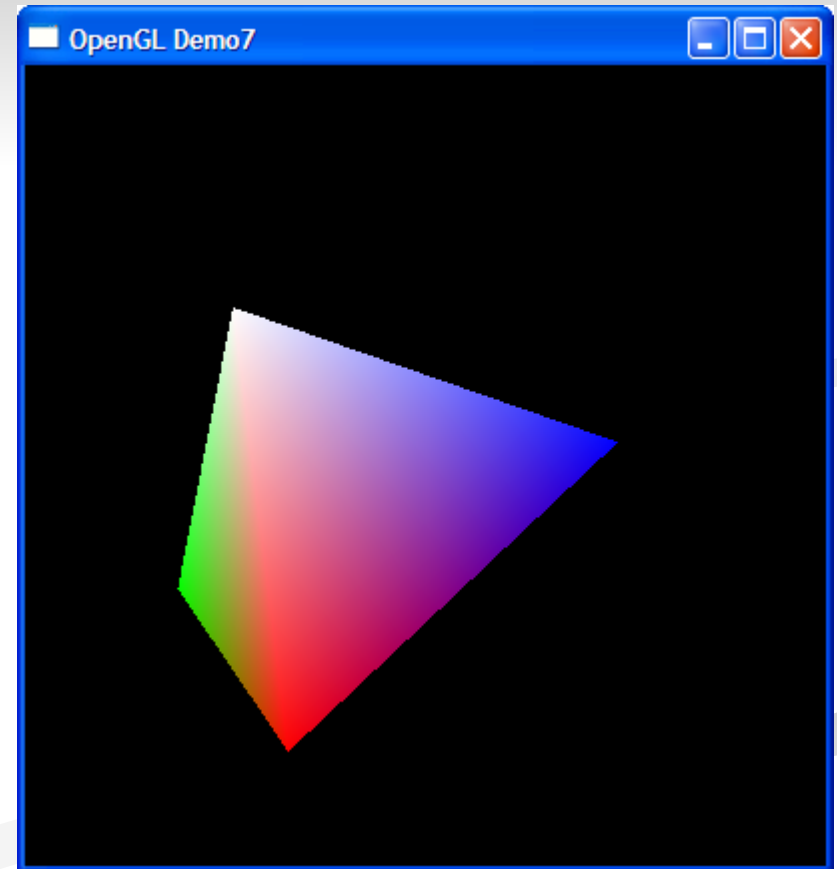
```
glEnableClientState(GL_VERTEX_ARRAY);
glEnableClientState(GL_COLOR_ARRAY);
glVertexPointer(4, GL_DOUBLE,
    sizeof(points[0]), points[0].vertex);
glColorPointer(3, GL_FLOAT,
    sizeof(points[0]), points[0].color);

glBegin(GL_TRIANGLES);
    glVertexElement(0);
    glVertexElement(1);
    glVertexElement(2);

    glVertexElement(0);
    glVertexElement(1);
    glVertexElement(3);

    glVertexElement(1);
    glVertexElement(2);
    glVertexElement(3);

    glVertexElement(2);
    glVertexElement(0);
    glVertexElement(3);
glEnd();
```



# Много елементи (1/3)

```
void glDrawElements (  
    GLenum mode,  
    GLsizei count,  
    GLenum type,  
    const GLvoid *indices)
```

# Много елементи (2/3)

**mode :**

**GL\_TRIANGLES, GL\_LINES, GL\_POINTS, ...**

**count :**

**Брой елементи за визуализация.**

**type :**

**GL\_UNSIGNED\_BYTE  
GL\_UNSIGNED\_SHORT  
GL\_UNSIGNED\_INT**

**indices :**

**Масив от индексите на елементите.**



# Много елементи (3/3)

```
glBegin(mode) ;  
for (int i = 0; i < count; i++)  
    glVertexElement(indices[i]) ;  
glEnd() ;
```

# Много елементи в диапазон (1/1)

```
void glDrawRangeElements (  
    GLenum mode,  
    GLsizei start,  
    GLsizei end,  
    GLsizei count,  
    GLenum type,  
    const GLvoid *indices)
```

# Много елементи в диапазон (2/2)

**mode:**

**GL\_TRIANGLES, GL\_LINES, GL\_POINTS, ...**

**start, end:**

**Минимален и максимален индекс в масива с индексите за визуализация.**

**count:**

**Брой елементи за визуализация.**

**type:**

**GL\_UNSIGNED\_BYTE  
GL\_UNSIGNED\_SHORT  
GL\_UNSIGNED\_INT**

**indices:**

**Масив от индексите на елементите.**

# Много последователни елементи (1/2)

```
void glDrawArrays (GLenum mode,  
GLint first, GLsizei count)
```

# Много последователни елементи (2/2)

```
glBegin(mode) ;  
for (int i = 0; i < count; i++)  
    glVertexElement(first + i);  
glEnd() ;
```

# Повече последователни елементи

```
void glMultiDrawArrays (  
    GLenum mode,  
    GLint *first, GLsizei *count  
    GLsizei primcount)
```

# Едновременна настройка

```
void glInterleavedArrays (  
    GLenum format,  
    GLsizei stride,  
    const GLvoid *pointer)
```

# Видове масиви

GL\_V2F

GL\_V3F

GL\_C4UB\_V2F

GL\_C4UB\_V3F

GL\_C3F\_V3F

GL\_N3F\_V3F

GL\_C4F\_N3F\_V3F

GL\_T2F\_V3F

GL\_T4F\_V4F

GL\_T2F\_C4UB\_V3F

GL\_T2F\_C3F\_V3F

GL\_T2F\_N3F\_V3F

GL\_T2F\_C4F\_N3F\_V3F

GL\_T4F\_C4F\_N3F\_V4F



# Пример 1

Например `GL_C3F_V3F` означава:

- `Color` с 3 `GLfloat` елемента;
- `Vertex` с 3 `GLfloat` елемента.

# Vertex Buffer Object (VBO)

```
void glGenBuffers(GLsizei n,  
                 GLuint* buffers)
```

```
void glDeleteBuffers(GLsizei n,  
                    const GLuint* buffers)
```

```
GLboolean glIsBuffer(GLuint  
                    buffer)
```

# Използване на VBO

```
void glBindBuffer(  
    GLenum target,  
    GLuint buffer)
```

**target:**

**GL\_ARRAY\_BUFFER** - Vertex атрибути;  
**GL\_ELEMENT\_ARRAY\_BUFFER** - индекси;

**buffer:**

**Име (номер) на буфера с данни.**

# Зареждане на данни във VBO

```
void glBufferData(  
    GLenum target,  
    GLsizei size, const void* data,  
    GLenum usage)
```

```
void glBufferSubData(...)
```

target:

GL\_ARRAY\_BUFFER - Vertex атрибути;

GL\_ELEMENT\_ARRAY\_BUFFER - индекси;

usage:

GL\_STATIC\_DRAW, GL\_STATIC\_READ, GL\_STATIC\_COPY

GL\_DYNAMIC\_DRAW, GL\_DYNAMIC\_READ, GL\_DYNAMIC\_COPY

GL\_STREAM\_DRAW, GL\_STREAM\_READ, GL\_STREAM\_COPY

# Промяна на данни във VBO

```
void* glMapBuffer(  
    GLenum target,  
    GLenum access)
```

```
GLboolean glUnmapBuffer(GLenum target)
```

target:

GL\_ARRAY\_BUFFER - Vertex атрибути;

GL\_ELEMENT\_ARRAY\_BUFFER - индекси;

access:

GL\_READ\_ONLY - само за четене;

GL\_WRITE\_ONLY - само за запис;

GL\_READ\_WRITE - за четене и запис.

# Пример 2 (1/2)

```
GLuint vboIdVertex = 0;  
  
// Създаваме VBO  
glGenBuffers(1, &vboIdVertex);  
  
// Активиране на VBO за използване  
glBindBuffer(GL_ARRAY_BUFFER, vboIdVertex);  
  
// Зареждаме данните във VBO  
glBufferData(GL_ARRAY_BUFFER,  
             ...dataSize..., ...data..., ...usage...);
```

# Пример (2/2)

```
// Преминаваме в режим на работа с VBO
glBindBuffer(GL_ARRAY_BUFFER, vboIdVertex);
glEnableClientState(GL_VERTEX_ARRAY);

// Последният параметър е отместване, а на указател!
glVertexPointer(3, GL_FLOAT, 0, 0);

glDrawArrays(GL_QUADS, 10, 5);
// Ако имаме свързан и GL_ELEMENT_ARRAY_BUFFER, то
// последният параметър на следващата ф-я е отместване!
// glDrawElements(GL_QUADS, 24, GL_UNSIGNED_BYTE, 0);

glDisableClientState(GL_VERTEX_ARRAY);

// Връщане в нормалния режим на работа с масиви
glBindBuffer(GL_ARRAY_BUFFER, 0);
```

# OpenGL Изграждане на Геометрични обекти 2

**Въпроси?**