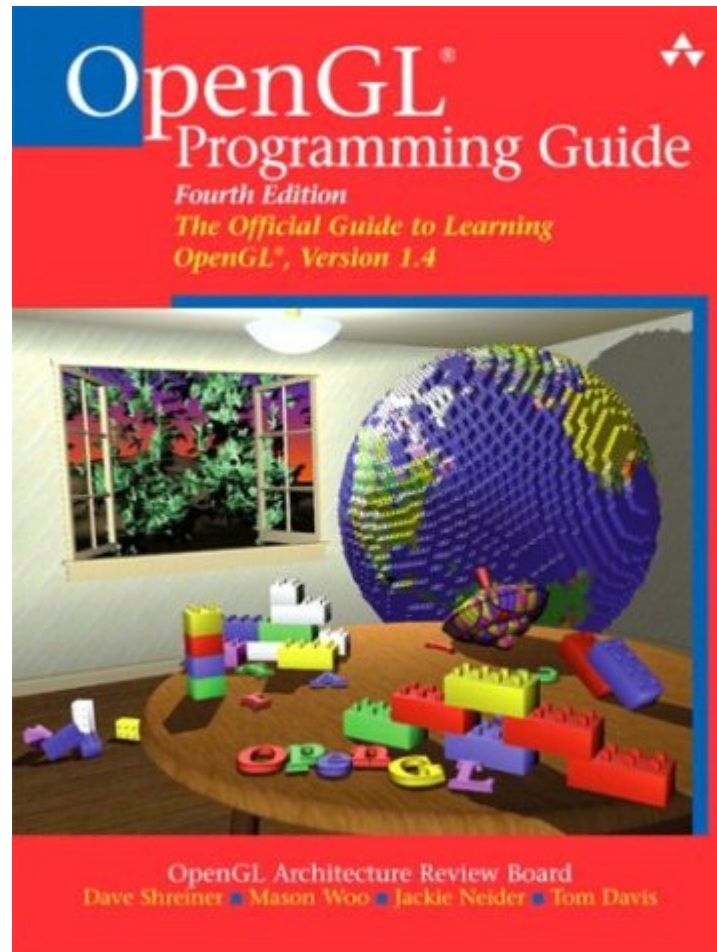


Kapitel 21: OpenGL

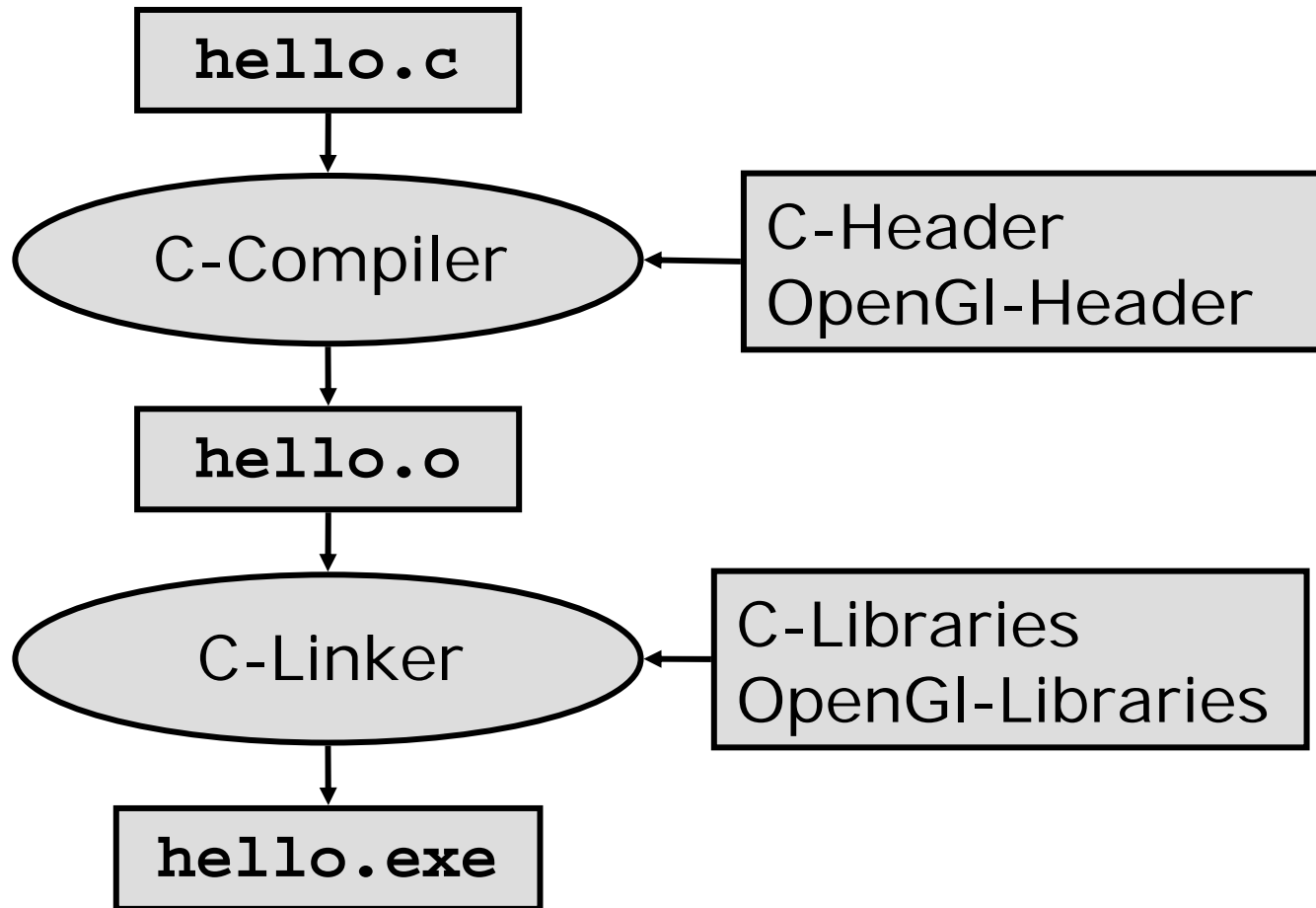
OpenGL Programming Guide



OpenGL

- 200 Befehle in OpenGL Library
(geometrische Primitive, Attribute)
`glColor3f(1.0,0.0,0.0);`
`glRotatef(30.0,0.0,0.0,1.0);`
- 50 Befehle in OpenGL Utility Library
(NURBS, Projektionsmatrizen)
`gluPerspective(60.0,1.5,1.0,20.0);`
- 30 Befehle in OpenGL Utility Toolkit
(Fenstersystem, Objekte)
`glutInitWindowPosition(100,100);`
`glutWireCube(2.0);`

Entwicklungszyklus



Programmaufbau

```
#include <GL/glut.h>
#include <stdlib.h>
void init(void){...}
void display(void){...}
void reshape(int w, int h){...}
int main(int argc, char ** argv){
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB)
    glutInitWindowSize(800, 600);
    glutInitWindowPosition(0,0);
    glutCreateWindow("Hello World");
    init();
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutMainLoop();
}
```

Datentypen

	Datentyp	OpenGL	C-Äquivalent
b	8-bit integer	GLbyte	signed char
s	16-bit integer	GLShort	short
i	32-bit integer	GLint	int
f	32-bit floating point	GLfloat	float
d	64-bit floating point	GLdouble	double
ub	8-bit unsigned int	GLubyte	unsigned char
us	16-bit unsigned short	GLushort	unsigned short
ui	32-bit unsigned int	GLuint	unsigned int

```
glVertex2i(5,3);
```

```
glVertex2f(5.0,3.0);
```

```
glVertex3f(5.0,3.0,1.0);
```

Parameterübergabe

Übergabe von 3 Zahlen:

```
glColor3f(1.0,0.8,0.8);
```

```
glColor3d(1.0,0.8,0.8);
```

Übergabe von Vektor:

```
GLfloat v[] = {1.0, 0.8, 0.8};
```

```
glColor3fv(v);
```

Zustandsmaschine

Attribute bleiben gültig bis auf Widerruf:

...

```
glColor3f(1.0,1.0,0.0);
```

```
/* ab jetzt in gelb zeichnen */
```


Geometrische Primitive

```
glBegin(GL_POINTS)... .. glEnd();
```

```
GL_LINES
```

```
GL_LINE_STRIP
```

```
GL_LINE_LOOP
```

```
GL_TRIANGLES
```

```
GL_TRIANGLE_STRIP
```

```
GL_TRIANGLE_FAN
```

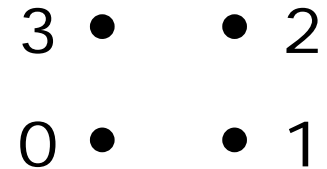
```
GL_QUADS
```

```
GL_QUAD_STRIP
```

```
GL_POLYGON
```

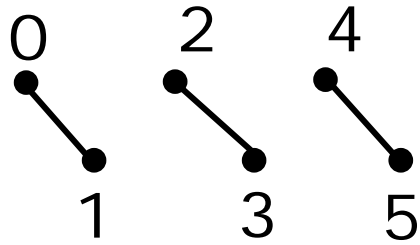
Punkte

```
glBegin(GL_POINTS)
    glVertex3f(0.25, 0.25, 0.0);
    glVertex3f(0.75, 0.25, 0.0);
    glVertex3f(0.75, 0.75, 0.0);
    glVertex3f(0.25, 0.75, 0.0);
glEnd();
```

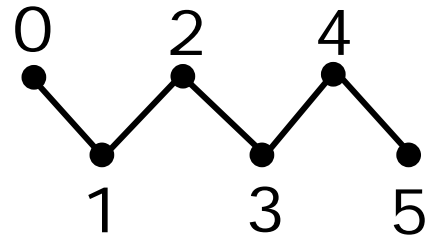


3 • • 2
0 • • 1

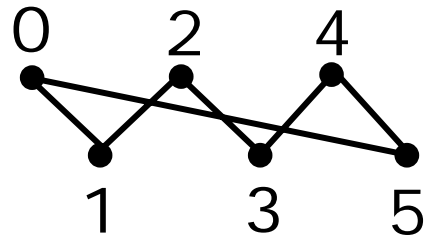
Linien



GL_LINES

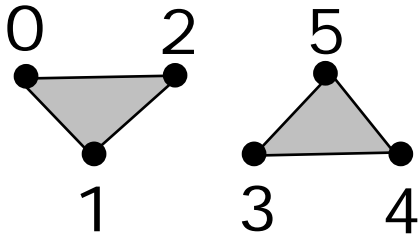


GL_LINE_STRIP

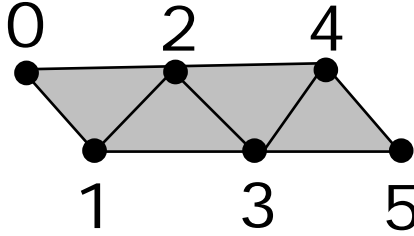


GL_LINE_LOOP

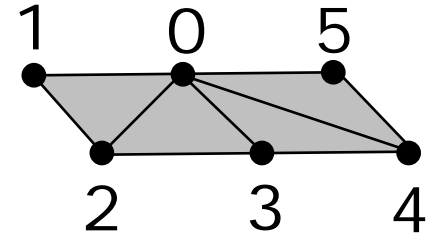
Polygone



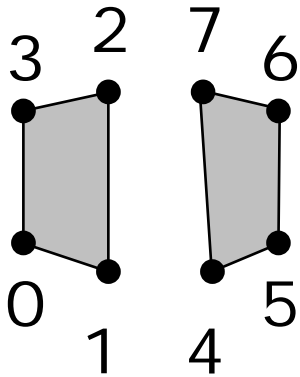
GL_TRIANGLES



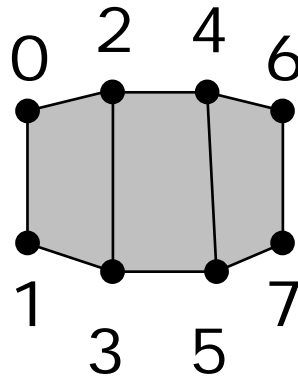
GL_TRIANGLE_STRIP



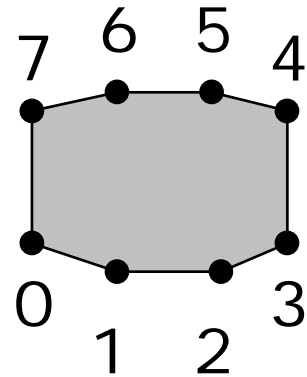
GL_TRIANGLE_FAN



GL_QUADS



GL_QUAD_STRIP



GL_POLYGON

Viewing Pipeline

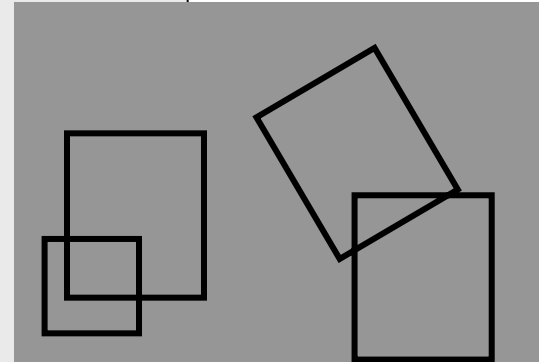
- Transformationsmatrix
für Projektion der Szene
- Transformationsmatrix
für Transformation der Objekte
- Graphische Objekte

Matrix-Operationen

```
glMatrixMode(GL_PROJECTION);  
glMatrixMode(GL_MODEL_VIEW);  
  
GLfloat M[][]={ {1.0,0.0,0.0,0.0},  
                 {0.0,1.0,0.0,0.0},  
                 {0.0,0.0,1.0,0.0},  
                 {5.0,7.0,2.0,1.0}};  
  
glMultMatrix(M);  
glTranslatef(5.0,7.0,2.0);  
  
glPushMatrix();  
glPopMatrix();
```

Transformation eines Quadrats

```
glMatrixMode(GL_MODELVIEW);  
glLoadIdentity ();  
glRotatef(30, 0, 0, 1);  
glTranslatef(2.0, -0.5, 0.0);  
glScale(1.5,2.0,1.0);  
glBegin(GL_Polygon)  
    glVertex3f(0.25, 0.25, 0.0);  
    glVertex3f(0.75, 0.25, 0.0);  
    glVertex3f(0.75, 0.75, 0.0);  
    glVertex3f(0.25, 0.75, 0.0);  
glEnd();
```



Projektion mit glOrtho

```
glMatrixMode (GL_PROJECTION);  
glLoadIdentity ();  
glOrtho(-4,4,-3,3,-5.0,5.0);  
glMatrixMode(GL_MODELVIEW);  
glLoadIdentity();  
glRotatef(20,1,0,0);  
glRotatef(-20,0,1,0);  
glutWireCube(1);
```


Projektion mit glFrustum

```
glMatrixMode(GL_PROJECTION);  
glLoadIdentity();  
glFrustum(-0.8,0.8,-0.6,0.6,1,20);  
glMatrixMode(GL_MODELVIEW);  
glLoadIdentity();  
glTranslatef(0,0,-4);  
glRotatef(20,1,0,0);  
glRotatef(-20,0,1,0);  
glutWireCube(1);
```

Projektion mit gluPerspektive

```
glMatrixMode (GL_PROJECTION);  
glLoadIdentity ();  
gluPerspective(60, 1.33, 1, 20);  
glMatrixMode(GL_MODELVIEW);  
glLoadIdentity();  
glTranslatef(0,0,-4);  
glRotatef(20,1,0,0);  
glRotatef(-20,0,1,0);  
glutWireCube(1);
```

Manipulation durch glLookAt

```
glMatrixMode (GL_PROJECTION);  
glLoadIdentity ();  
gluPerspective(60, 1.33, 1, 20);  
glMatrixMode(GL_MODELVIEW);  
glLoadIdentity();  
gluLookAt (1,1,3,0,0,0,0,1,0);  
glutWireCube(1);
```

Reshape

```
void reshape(int w, int h) {  
    GLfloat p = (GLfloat) w / (GLfloat) h;  
    glViewport(0, 0, w, h);  
    glMatrixMode(GL_PROJECTION);  
    glLoadIdentity();  
    if (p > 1.0)  
        glFrustum( -p, p, -1.0, 1.0, 1, 20);  
    else  
        glFrustum(-1.0, 1.0, -1/p, 1/p, 1, 20);  
}
```

Programmbeispiele

hello.exe

[hello.c](#)

wire-cube.exe

[wire-cube.c](#)

wire-prop-cube.exe

[wire-prop-cube.c](#)

click.exe

[click.c](#)

key.exe

[key.c](#)

planet.exe

[planet.c](#)

smooth.exe

[smooth.c](#)

list.exe

[list.c](#)

bezier-curve.exe

[bezier-curve.c](#)

bezier-surface.exe

[bezier-surface.c](#)

teapot-rotate.exe

[teapot-rotate.c](#)

atlantis.exe

[dolphin.c](#)

Aufgabe

Roboter mit 3 Freiheitsgraden

