

Pengenalan OpenGL

OpenGL adalah suatu graphic library yang sebagian bersifat open source, dipakai pada banyak platform (windows, linux) dan dapat digunakan pada berbagai jenis compiler seperti C++ atau Delphi.

OpenGL bukanlah bahasa pemrograman tetapi merupakan suatu *Application Programming Interface (API)*.

Sintax Perintah OpenGL

Sintaks perintah OpenGL mengikuti aturan penulisan dari *library* dimana fungsi tersebut berasal, format penulisan fungsi OpenGL adalah :

<awalan *library*><perintah><optional jumlah argumen><optional tipe argumen>

Semua perintah OpenGL menggunakan awalan *gl* diikuti dengan huruf kapital pada setiap kata membentuk nama perintah (sebagai contoh *glClearColor*). Untuk mendefinisikan konstanta diawali dengan *GL_*, dengan menggunakan huruf kapital dan garis bawah untuk memisahkan kata (seperti *GL_POLY_STIPPLE*). Terkadang beberapa huruf dan angka ditambahkan pada akhir perintah (seperti *3f* pada *glVertex3f*). Dalam hal ini angka 3 menunjukkan berapa banyak argumen yang harus ada pada perintah tersebut dan akhiran huruf *f* menunjukkan jenis datanya yaitu *floating*. Sebagai contoh pada dua perintah berikut ini :

```
glVertex3i(1,0,-2);
```

```
glVertex3f(1.0, 0.0, -2.0);
```

adalah sama yaitu meletakkan titik di layar pada koordinat $x = 1$, $y = 0$ dan $z = -2$, perbedaannya yaitu pada perintah pertama menspesifikasikan titik dengan tipe data *integer 32-bit*, sedangkan yang kedua dengan tipe data *single precision floating point*.

Beberapa perintah OpenGL menambahkan perintah huruf akhir *v* yang menunjukkan bahwa perintah tersebut menggunakan pointer ke *array/vektor*. Di bawah ini contoh perbedaannya.

```
float color_array[]={1.0,0.0,0.0}
```

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

```
glColor3fv(color_array);
```

Library yang Berhubungan dengan OpenGL

OpenGL menyediakan set perintah untuk menggambar dan semua penggambaran yang lebih tinggi tingkatnya harus dilakukan dengan mengambil fungsi dasar dari perintah ini. Maka dari itu dapat dibuat *library* itu sendiri di atas program OpenGL yang mempermudah pemrograman lebih lanjut. Fungsi asli dari OpenGL sendiri selalu diawali dengan *gl* yang terdapat pada library *opengl32.dll* dan file header *gl.h*. Sedangkan beberapa *library* yang telah ditulis untuk menyediakan fungsi-fungsi tambahan pada OpenGL adalah :

1. OpenGL Utility Library (GLU) yang didalamnya terdapat sejumlah rutin yang menggunakan level bawah dari perintah OpenGL. Rutin-rutin ini mempunyai awalan *glu*. Library ini digunakan sebagai bagian dari implementasi OpenGL.
2. OpenGL Extension untuk X-Windows yang menyediakan fungsi untuk menciptakan OpenGL context dan mengasosiasikannya dengan mesin yang menggunakan X-Windows. Rutin-rutin ini mempunyai awalan *glx*.
3. Auxiliary atau aux library terdapat pada library *glaux.lib* dan file header *glaux.h*. Perintah yang akan digunakan selalu menggunakan awalan *aux*
4. *OpenGL Utility Toolkit (GLUT)* adalah *toolkit* untuk sistem *windows* yang ditulis oleh Mark Kilgard untuk menyembunyikan perintah *API* sistem *windows* yang kompleks.

Menggambar Objek Geometri

Pada OpenGL ada dua dasar operasi gambar yaitu membersihkan *windows* dan menggambar objek geometri termasuk titik, garis dan poligon.

Membersihkan *Windows*

Menggambar pada layar komputer berbeda dengan menggambar pada kertas putih yang dari pabriknya sudah berwarna putih. Pada komputer, *memory* untuk menampilkan gambar biasanya diisi dengan gambar yang berasal dari perintah gambar paling akhir, jadi perlu dibersihkan dengan warna latar belakang sebelum digambar lagi. Warna latar belakang yang dipilih tergantung dari aplikasi yang akan dibuat. Sintaks *glClearColor(Glclamp red, Glclamp green, Glclamp blue, Glclamp alpha)* digunakan untuk memilih warna, yang akan digunakan untuk membersihkan latar belakang dalam *mode RGBA*. Selanjutnya perintah *glClear(Glbitfield mask)* digunakan untuk membersihkan *buffer* yang dispesifikasikan dengan warna yang telah ditentukan. Contoh berikut ini perintah yang digunakan untuk membersihkan layar latar belakang dengan warna hitam dan *buffer* apa yang akan dibersihkan. Dalam hal ini, *buffer* warna yang akan dibersihkan karena *buffer* warna merupakan tempat gambar disimpan.

```
glClearColor(0.0,0.0,0.0,0.0);  
glClear(GL_COLOR_BUFFER_BIT);
```

Spesifikasi Warna

Pada OpenGL mendeskripsikan objek dengan warna objek adalah proses yang berjalan sendiri-sendiri. Karena pada umumnya seorang *programmer* akan mengatur warna terlebih dahulu lalu menggambar objek. Sebelum warna diubah maka semua objek yang digambar sesudah perintah tersebut akan menggunakan warna terakhir yang terdapat pada *coloring scheme*.

Untuk warna digunakan perintah *glColor3f()*, jika lebih dari tiga maka argumen keempat adalah *alpha* yang akan dijelaskan pada bagian *blending* sebagai salah satu efek yang dipunyai OpenGL. Contoh berikut menunjukkan urutan langkah dalam proses spesifikasi warna sebelum objek digambar.

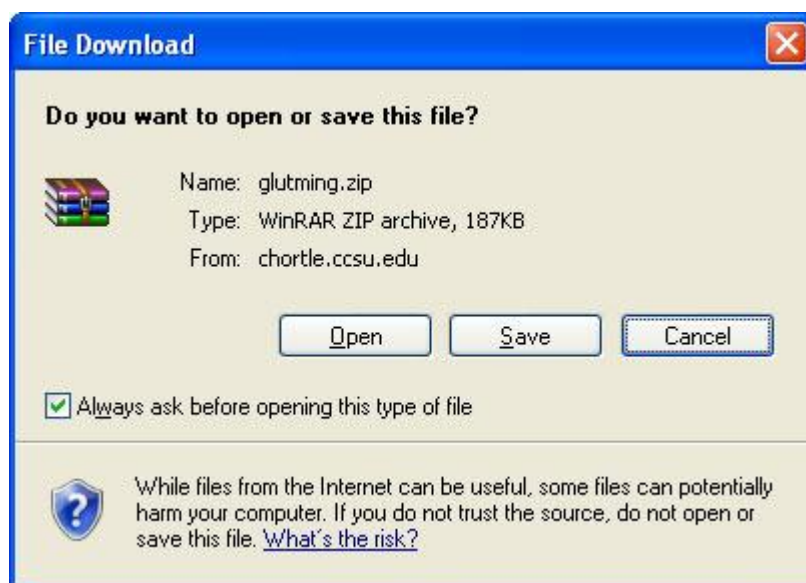
```
glColor3f(0.0,1.0,0.0);           //setting warna  
draw_object(A);                   //gambar objek A
```

Memaksa Proses Menggambar Sampai Selesai

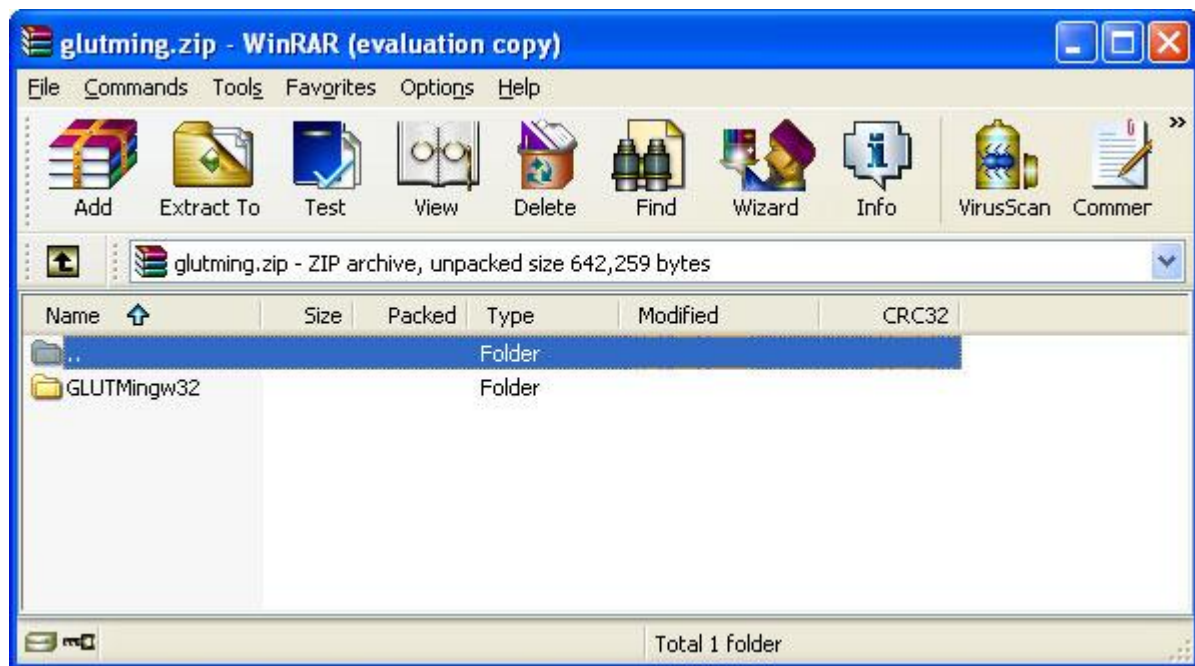
Kebanyakan sistem grafik modern sudah menggunakan sistem *graphics pipeline*. Dimana CPU utama memberikan *issue* perintah menggambar dan hardware lain yang melakukan *transformasi, clipping, shading, texturing* dan lain-lain. Pada arsitektur yang demikian, proses tidak dilakukan pada satu komputer karena setiap komputer mempunyai tugas sendiri. CPU utama tidak harus menunggu proses pada masing-masing komputer tadi selesai, tapi bisa dengan memberikan *issue* perintah gambar yang berikutnya. Untuk inilah OpenGL menyediakan perintah *glFlush()* yang memaksa *client* untuk segera mengirim paket *network* walaupun belum penuh. Program sebaiknya ditulis menggunakan perintah ini karena *glFlush()* tidak memaksa proses gambar untuk selesai tetapi memaksa proses gambar untuk segera dieksekusi, sehingga dijamin semua perintah gambar yang sebelumnya akan segera dieksekusi dalam suatu waktu tertentu.

Instalasi OpenGL

Untuk menjalankan OpenGL dengan GLUT, ada tiga file yang harus ditempatkan dengan baik disetiap direktori, dan file-file tersebut ditempatkan dalam 1 file zip. File-file tersebut dapat didownload di <http://chortle.ccsu.edu/Bloodshed/glutming.zip>.

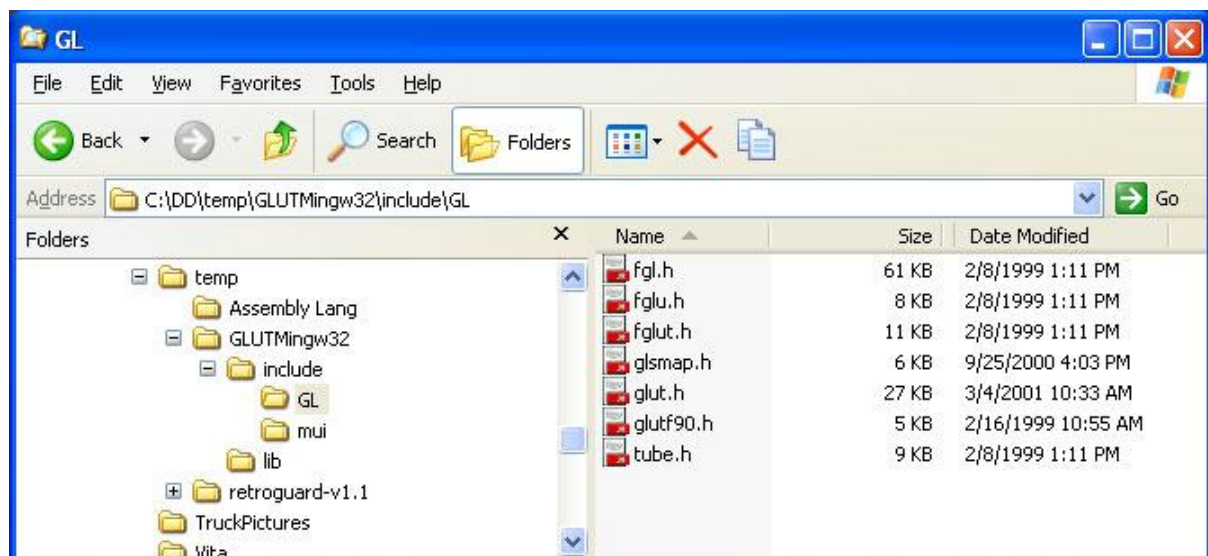


Lakukan ekstraksi dari file hasil download di direktori yang ditentukan.

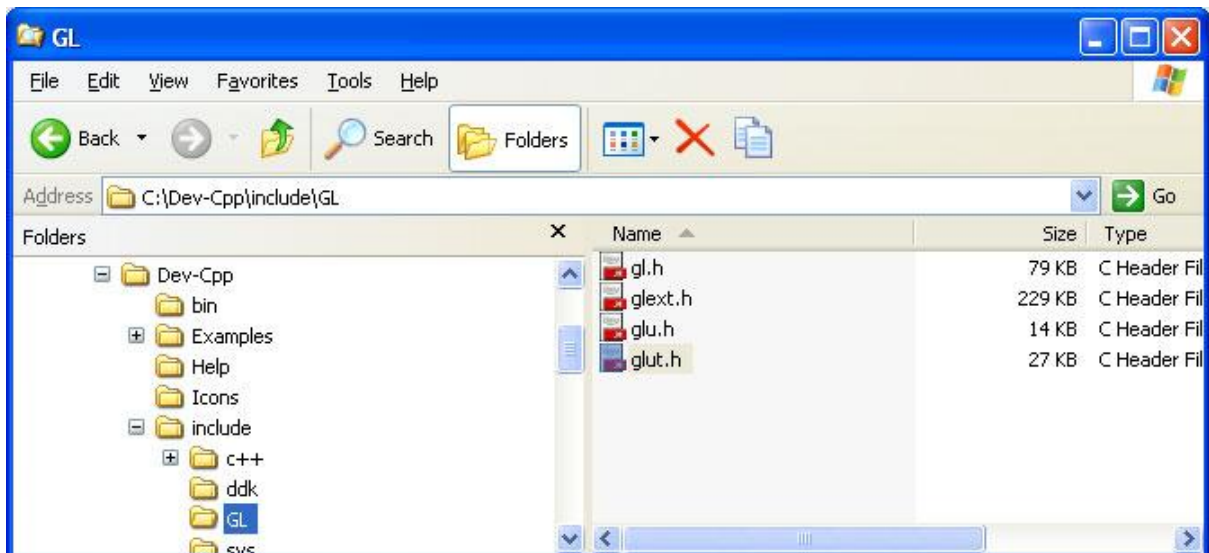


Dari hasil ekstraksi tersebut, terdiri dari beberapa file. Hanya tiga file yang dibutuhkan.

Glut.h → copy file ini ke C:\Dev-cpp\include\GL (atau didirektori dimana Dev C++ diinstalasi.), seperti dicontohkan dalam gambar berikut ini

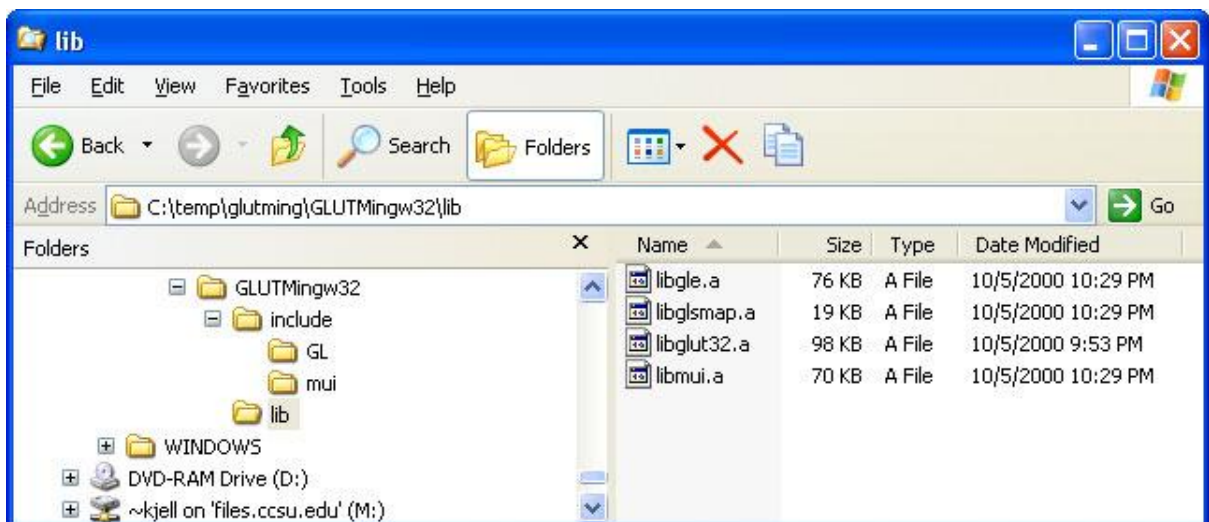


Ke direktori C:\Dev-cpp\include\GL

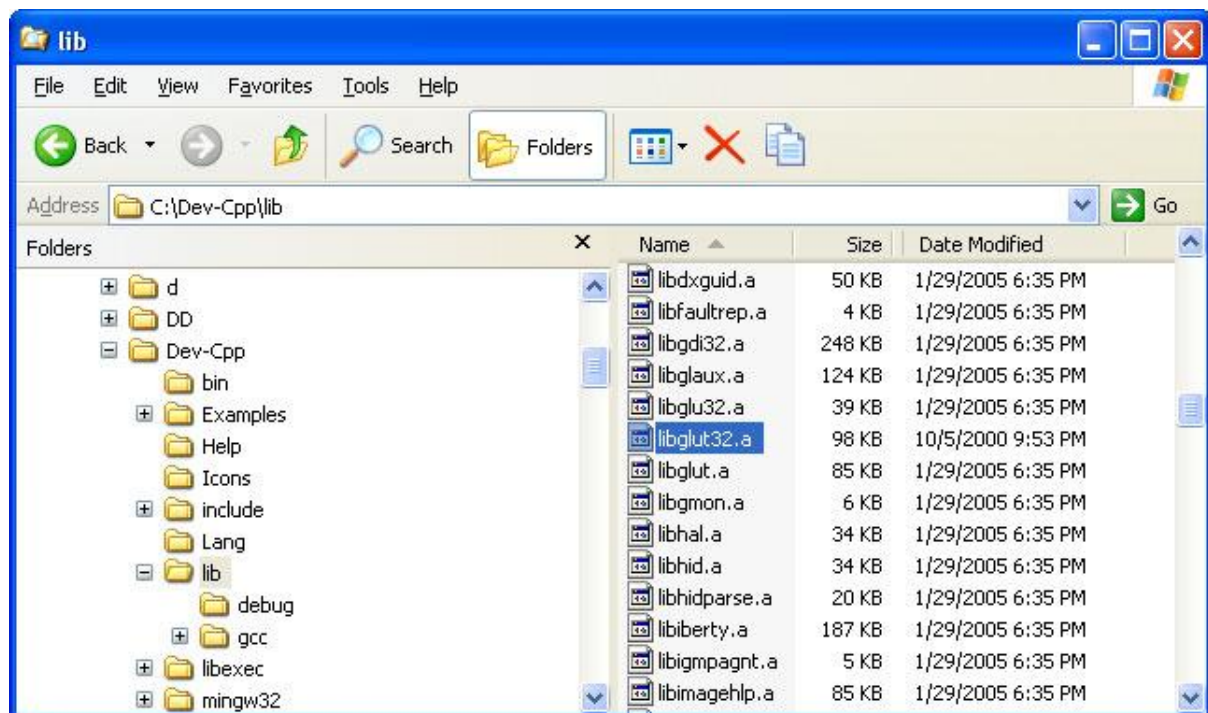


Libglut32.a → copykan file ini ke direktori C:\Dev-Cpp\lib

Dimungkinkan terdapat versi terbaru dari file tersebut. *Replace* versi tersebut dengan yang telah diekstraksi (dimungkinkan program tidak terhubung dengan baik).



Disimpan didirektori yang sudah ditentukan



Glut32.dll → pindahkan file tersebut ke C:\\Windows\\System32

Penggambaran Primitif Grafik Dasar

OpenGL menyediakan banyak fungsi untuk kebanyakan grafik primitif termasuk titik, garis dan lingkaran. Masing-masing perintah atau fungsi dalam OpenGL mempunyai format dan struktur yang sama. Ini memudahkan untuk mengetahui bagaimana jalannya fungsi, argumen yang dibutuhkan untuk melakukannya dan format tipe datanya. Pada tabel 1 menunjukan contoh perintah OpenGL yang digunakan, dan tabel 2 menunjukan format fungsi OpenGL.

Tabel 1. Contoh-contoh Perintah dalam OpenGL

Perintah	Arti	Keterangan
<code>glVertex2i(x,y);</code>	Lokasi titik berada (x,y)	Tipe argumennya adalah integer dan 2D (x,y)
<code>glVertex2f(x,y);</code>	Lokasi titik berada (x,y)	Tipe argumennya adalah float dan 2D (x,y)
<code>glVertex3i(x,y,z);</code>	Lokasi titik berada (x,y,z)	Tipe argumennya adalah integer dan 3D (x,y,z)
<code>glVertex3f(x,y,z);</code>	Lokasi titik berada (x,y,z)	Tipe argumennya adalah float dan 3D (x,y,z)
<code>glClearColor(R,G,B,α);</code>	Warna latar belakang	Empat komponen warna RGBA
<code>glColor3f(R,G,B)</code>	Warna latar muka	Tiga komponen warna (RGB)

<code>glColor4f(R,G,B,α)</code>	Warna latar muka	Empat komponen warna (RGBA)
<code>glBegin(GL_POINTS);</code>	Titik	Objek primitif
<code>glBegin(GL_LINES);</code>	Garis	Objek primitif
<code>glBegin(GL_LINE_STRIP);</code>	Poligaris	Objek primitif
<code>glBegin(GL_LINE_LOOP);</code>	Poligaris tertutup	Objek primitif
<code>glBegin(GL_TRIANGLES);</code>	Segitiga	Objek primitif
<code>glBegin(GL_TRIANGLE_STRIP);</code>	Segitiga	Objek primitif
<code>glBegin(GL_TRIANGLE_FAN);</code>	Segitiga	Objek primitif
<code>glBegin(GL_QUADS);</code>	Segiempat	Objek primitif
<code>glBegin(GL_QUADS_STRIP);</code>	Segiempat	Objek primitif
<code>glBegin(GL_LINE_STIPPLE);</code>	Garis putus-putus	Objek primitif
<code>glBegin(GL_POLY_STIPPLE);</code>	Poligon dengan pola tertentu	Objek primitif

Tabel 2. Format Fungsi OpenGL

Suffix	Type Data	C++ atau C	OpenGL
b	Integer 8-bit	Signed char	GLbyte
s	Integer 16-bit	Short	GLshort
i	Integer 32-bit	Int atau long	GLint, GLsizei
f	Float 32-bit	float	GLfloat
d	Float 64-bit	Double	GLdouble

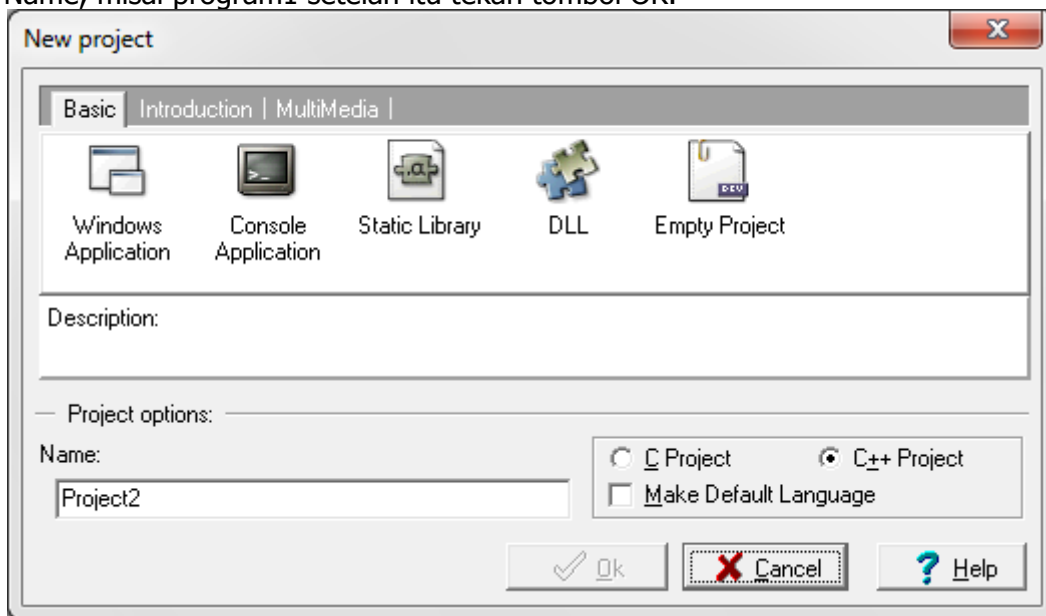
Memulai DevC++ dan OpenGL

Pada bagian ini akan diterangkan langkah-langkah dalam pembuatan project, setting untuk OpenGL, serta pembuatan program dengan menggunakan DevC++

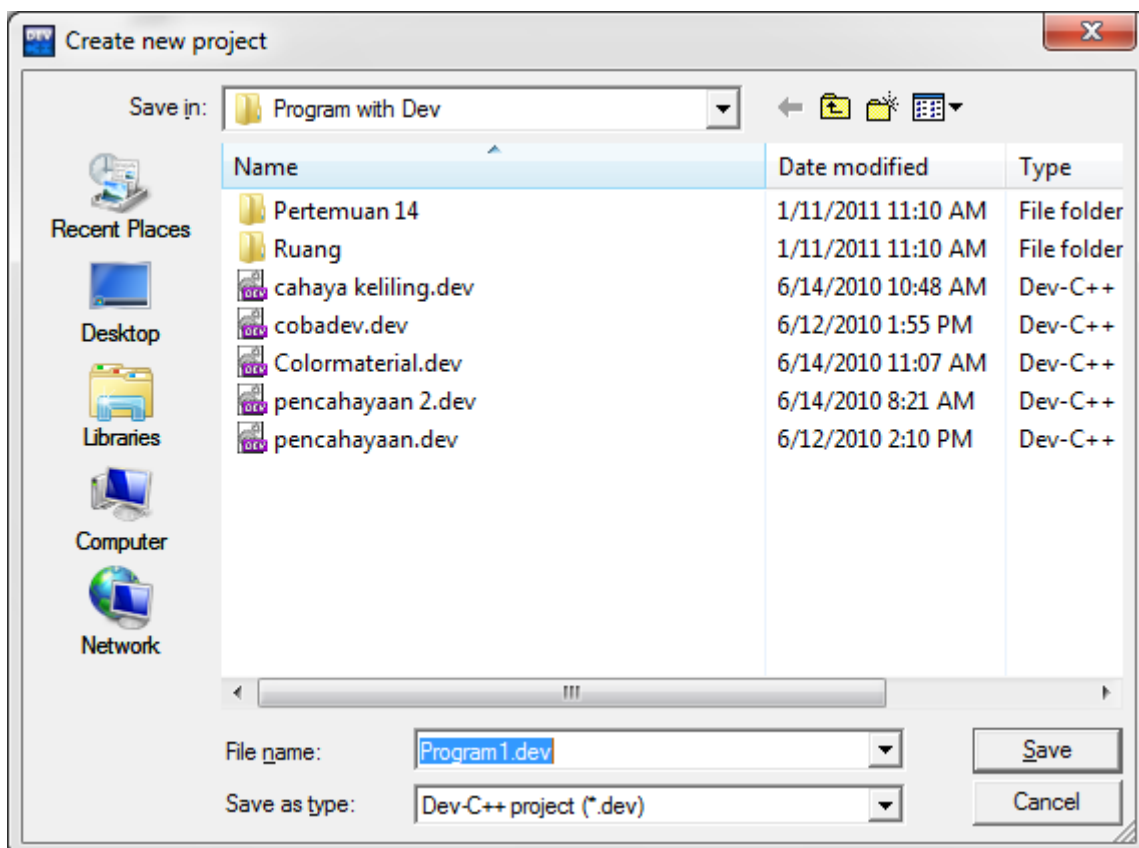
Membuat Project Baru

1. Pilih **Start** → **Program** → **Dev C++**
2. Pilih menu **File** → **New**, kemudian akan tampil kotak dialog seperti pada gambar berikut. Kemudian pilih tab **Project** → **Console Application**, tentukan lokasi file

yang akan dibuat, misal : D:\Grafika\, kemudian berikan nama proyek pada Project Name, misal program1 setelah itu tekan tombol OK.

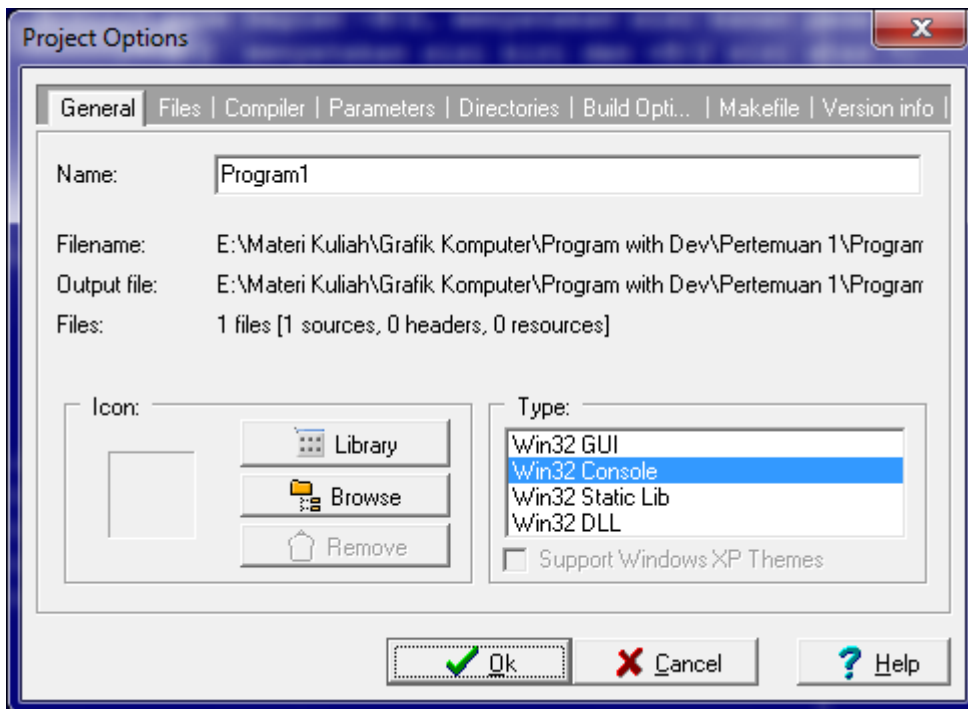


3. Selanjutnya akan muncul kotak dialog *create new project*. Tentukan tempat penyimpanannya seperti gambar berikut.



4. Setelah itu pilih "File/New/Source File", kemudian pada panel berikut "Add to Project" klik "yes". Save file tersebut berakhiran .c agar tidak menimbulkan masalah.

5. Setting library di "project/project option"



6. Pilih "parameter". Klik tombol "add library or project" kemudian tambahkan file berikut :

- a. ../lib/libopengl32.a
- b. ../lib/libglu32.a
- c. ../lib/libglut32.a

Program 1 (layar.cpp)

```
/* Program layar.cpp
 * Pengenalan OpenGL
 */
#include <windows.h>
#include <GL\glut.h>

void display(void)
{
    //membersihkan buffer
    glClear(GL_COLOR_BUFFER_BIT);
    glFlush();
}

void init(void) {
    //membuat background warna merah
    glClearColor(1.0,0.0,0.0,0.0);
    //set viewport dengan nilai 320 x 240, inisialisasi window size
    glViewport(0,0,320,240);
    // set the 2D clipping area
    gluOrtho2D(0.0, 160.0,0.0, 120.0);
}

/* Deklarasi window size, position, dan display mode(single buffer and RGBA).
 * Pemanggilan rutin inisialisasi.
```

```

    * Memanggil fungsi untuk menampilkan objek di layar
    */

int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize (320, 240);
    glutInitWindowPosition (200, 100);
    glutCreateWindow ("Program layar");
    init ();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}

```

Program 2 (Hallo.cpp)

```

/*
 * Program hallo.cpp
 * Pengenalan OpenGL
 */

#include <windows.h>
#include <GL/glut.h>

void display(void)
{
    /* Membersihkan semua pixel di buffer(clear all pixels) */
    glClear (GL_COLOR_BUFFER_BIT);

    /* Menggambar polygon (rectangle) (0.25, 0.25, 0.0) dan (0.75, 0.75, 0.0)
     * Objek berwarna hijau
     */
    glColor3f (0.0, 1.0, 0.0);
    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();

    /* Memulai proses penggambaran
     */
    glFlush ();
}

void init (void)
{
    // Membuat warna background

    glClearColor (0.0, 0.0, 0.0, 0.0);

    /* inisialisasi nilai tampilan (viewing values) */
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(0.0, 1.0, 0.0, 1.0, -1.0, 1.0);
}

/* Deklarasi window size, position, dan display mode(single buffer and RGBA).
 * Pemanggilan rutin inisialisasi.
 * Memanggil fungsi untuk menampilkan objek di layar
 */
int main(int argc, char** argv)
{
    glutInit(&argc, argv);

```

```

    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize (640, 480);
    glutInitWindowPosition (200, 100);
    glutCreateWindow ("Program Pertamaku");
    init ();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}

```

Program 3 Segi empat

```

#include <GL/glut.h>

const int    A = 500; /* panjang/lebar monitor window */
const float  B = 500; /* Batas clipping window */
const float  C = 200; /* panjang ukuran segiempat */

void myinit(void)
{
    glClearColor(0.7, 0.7, 0.7, 0.0); /* warna latar abu-abu */

    glMatrixMode(GL_PROJECTION);      /* coordinates: nyata */
    glLoadIdentity();                 /* posisi "clipping rectangle" */
    gluOrtho2D( -B/2, B/2, -B/2, B/2); /* pada bagian -B/2, menyatakan sisi kanan pada
+B/2, menyatakan sisi bawah */
    glMatrixMode(GL_MODELVIEW);       /* -B/2  menyatakan sisi kiri dan +B/2 sisi atas
*/
}

void display( void )
{
    glClear(GL_COLOR_BUFFER_BIT);      /* clear the window */

    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();

    glBegin(GL_POLYGON) ;              /* menggambar polygon */
        glColor3f ( 1.0, 0.3, 0.2);    /* draw in light red */
        glVertex2f( -C/2, -C/2 );      /* (x,y) */
        glVertex2f(  C/2, -C/2 );      /* (x,y) */
        glVertex2f(  C/2,  C/2 );      /* (x,y) */
        glVertex2f( -C/2,  C/2 );      /* (x,y) */
    glEnd();

    glFlush();
}

void main(int argc, char** argv)
{
    glutInit(&argc,argv);
    glutInitWindowSize( A, A );        /* A x A pixel screen window */

    glutInitDisplayMode( GLUT_RGB | GLUT_SINGLE);
    glutCreateWindow("Program pertama"); /* window title */
    glutDisplayFunc(display);          /* tell OpenGL main loop what */
    myinit();                          /* set attributes */

    glutMainLoop();                   /* pass control to the main loop */
}

```

Program 4 (garis.cpp)

```

/* Program garis.cpp
* Pengenalan OpenGL

```

```

*/
#include <windows.h>
#include <GL\glut.h>

void display(void)
{
    //membersihkan buffer
    glClear(GL_COLOR_BUFFER_BIT);
    //menggambar garis
    glBegin(GL_LINES);
        glColor3f(1.0,0.0,0.0); //merah
        glVertex2f(10.0,10.0);
        glColor3f(0.0,0.0,1.0); //biru
        glVertex2f(10.0,110.0);
        glColor3f(0.0,1.0,0.0); //hijau
        glVertex2f(100.0,110.0);
        glColor3f(1.0,1.0,0.0); //kuning
        glVertex2f(100.0,10.0);
    glEnd();
    glFlush();
}

void reshape(int w, int h){
    glViewport (0,0, (GLsizei) w, (GLsizei) h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0.0,160.0,0.0,120.0);
}

void init(void){
    //membuat background warna merah
    glClearColor(0.0,0.0,0.0,0.0);
    //set viewport dengan nilai 320 x 240, inisialisasi window size
    glViewport(0,0,320,240);
    // set the 2D clipping area
    gluOrtho2D(0.0, 160.0,0.0, 120.0);
    glPointSize(5.0);
}

/* Deklarasi window size, position, dan display mode(single buffer and RGBA).
 * Pemanggilan rutin inisialisasi.
 * Memanggil fungsi untuk menampilkan objek di layar
 */

int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize (320, 240);
    glutInitWindowPosition (200, 100);
    glutCreateWindow ("Program layar");
    init ();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}

```