KARADENİZ TEKNİK ÜNİVERSİTESİ
Bilgisayar Mühendisliği Bölümü
Bilgisayar Grafikleri Laboratuarı

OpenGL Uygulamaları

1. Giriş

Günümüzde yazılım ve donanımın gelişmesi ile birlikte bilgisayar grafikleri alanında oldukça önemli gelişmeler kaydedilmektedir. Bu gelişmelere paralel olarak yazılım geliştirme kütüphaneleri de ortaya çıkmaktadır. OpenGL (Open Graphic Library) de gelişmiş grafik uygulamaları için geliştirilen bir kütüphanedir.

OpenGL grafik kütüphanesi, grafik donanımına bir yazılım arayüzüdür. 2D ya da 3D nesnelerden oluşan hareket eden görüntüler üreten, interaktif, çok çeşitli bilgisayar platformlarında çalışabilecek programlar yaratmamıza olanak sağlar. OpenGL, donanım-bağımsız bir arayüzdür. Görüntüde bulunan nesneleri tanımlamak ve bu nesneler üzerinde gerek duyulan işlemleri gerçekleştirmek için yaklaşık 700 farklı komut içerir. (Bu komutlardan yaklaşık 650 tanesi saf OpenGL komutu ve yaklaşık 50 tanesi ise OpenGL Utility Library de bulunmaktadır [1].

OpenGL komut yorumlama modeli, istemci/sunucu (client/server) modelidir. Uygulama kodu (istemci), komut çağrılarını yapar. Bu komutlar, OpenGL (sunucu) tarafından yorumlanır ve işlenir. OpenGL, grafik programını koşan ve yarattığımız grafiği görüntüleyen bilgisayarlar farklı olduğunda bile etkili çalışacak şekilde tasarlanmıştır. Bu durum, istemci ve sunucunun farklı tip bilgisayarlar olabileceği bir bilgisayar ağı ortamında geçerlidir.

OpenGL ‘in donanım-bağımsız olmasının nedeni, pencere işlemlerini (windowing task-ekranda bir pencere oluşturmak gibi) yapan ya da kullanıcıdan girdi alan herhangi bir komutunun bulunmamasıdır. Belirtilen bu işleri gerçekleştirmek için işletim sisteminin mevcut özellikleri kullanılır. Ancak işletim sisteminde pencere işlemlerini gerçekleştirmek karmaşık işlemler içerdiğinden tüm bu işlevleri barındıran ve işletim sistemine özel olarak yazılmış GLUT (Graphic Library Utility) kütüphaneleri bulunmaktadır.

OpenGL, 3D nesneleri tanımlamak için yüksek-seviye komutlar içermez. Bunun yerine; nokta, doğru ve poligon gibi alt-seviye geometrik primitifleri kullanarak nesneleri tanımlamamıza olanak sağlar. Primitifler, bir ya da daha fazla vertex ile tanımlanır. Bir nokta, bir doğrunun başlangıç ve bitiş noktaları, bir poligonun iki kenarının kesiştiği köşe noktası, vertex ‘ ler yardımıyla belirlenir.

2. OpenGL

2.1.Neden OpenGL ?

OpenGL kullanarak grafikler oluşturmanın avantajları aşağıda sıralanmıştır.
Tüm OpenGL uygulamaları, işletim sistemi ne olursa olsun, OpenGL API uyumlu donanımlar üzerinde mükemmel görsel sonuçlar üretebilir.
Grafik donanımlarının yeni gelişmiş özellikleri, OpenGL tarafından, geliştirme mekanizması (extension mechanism) sayesinde kullanılabilir. Yani OpenGL, donanıma-özel, gelişmiş özellikleri kullanmak için API fonksiyonları içerebilir.
OpenGL temelli grafik uygulamaları, çok çeşitli sistemler üzerinde koşulabilir. (tüketici elektroniği – consumer electronics, PC, iş istasyonu – workstation, süper bilgisayarlar gibi)
OpenGL grafik kütüphanesi kullanılarak, çok daha az bir kod satırıyla daha yüksek performansa

sahip uygulamalar geliştirmek mümkündür.
OpenGL grafik kütüphanesine dair teknik bilgi içeren birçok kaynak mevcuttur.(internet, kitaplar, vs.)
Birçok programlama dili (C, C++, Fortran, Ada, Java gibi) OpenGL tabanlı uygulama geliştirmemize olanak sağlar.

2.2.Open GL Utility (GLUT)

OpenGL platformdan bağımsız olduğu için bazı işlemler bu kitaplık ile yapılamaz. Örneğin kullanıcıdan veri almak, bir pencere çizdirmek gibi işler hep kullanılan pencere yöneticisi ve işletim sistemine bağlıdır. Bu yüzden bir an için OpenGL'in platform bağımlı olduğu düşünülebilir. Çünkü çalışma penceresini her pencere yöneticisinde (her ortamda) farklı çizdirecek bir canlandırma programı yazmak demek her bilgisayarda çalışacak ayrı pencere açma kodu yazmak demektir. Bu ise OpenGL'in doğasına aykırıdır. Bu gibi sorunları aşmak için OpenGL Araç Kiti (GLUT - OpenGL Utility Toolkit) kullanılmaktadır.

GLUT, birçok işletim sistemine aktarılmış bir kitaplıktır. Amacı OpenGL programlarının pencerelerini oluşturmak, klavye ve fareden veri almak gibi ihtiyaçları karşılamaktır.

GLUT olmadan da OpenGL programlama yapılabilir, örneğin Linux'ta kullanılan X-Window sistemin kendi işlevleri kullanılarak pencere çizdirilebilir fakat bu kod sadece X-Window'da çalışır. Kod Windows'a götürülüp derlendiğinde çalışmaz, çünkü Windows'da X-Window işlevleri yoktur. Benzer şekilde Windows tabanlı işletim sistemlerinin de kendilerine has pencere oluşturma işlevleri vardır.

Bu yüzden bu deneyde GLUT kitaplığı kullanılarak klavye ve fare için işletim sisteminden bağımsız giris/çıkış işlemleri yapılması sağlanmıştır.

2.3.OpenGL Söz dizimi

Opengl komutları, gl öneki ile başlarlar. (örnek; glClearColor()). Benzer şekilde OpenGL tarafından tanımlı sabitler de GL_ öneki ile başlarlar ve kelimeler birbirinden _ ile ayrılacak şekilde büyük harfle yazılırlar. (örnek; GL_COLOR_BUFFER_BIT).

glColor3f komutundaki 3 sayısı da 3 parametre alacağı anlamına gelmektedir. f ise verilen parametrelerin float olacağı anlamına gelmektedir.

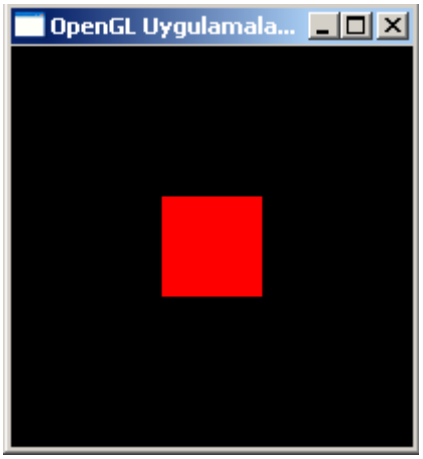
```
glVertex2i(1,3);   ya da
glVertex2f(1.0,3.0);   gibi
```

2.4.İlk OpenGL Programı

```
#include <GL/glut.h>
#include <stdlib.h>

void ayarlar(void)
{
    glClearColor(0.0,0.0,0.0,0.0);
    glShadeModel(GL_FLAT);
    glOrtho(-2.0, 2.0, -2.0, 2.0, -1.0, 1.0);

}
```

```
void gosterim(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(1.0, 0.0, 0.0);

    glBegin(GL_POLYGON);
       glVertex2f(-0.5, -0.5);
       glVertex2f(-0.5, 0.5);
       glVertex2f(0.5, 0.5);
       glVertex2f(0.5, -0.5);
    glEnd();
    glFlush();
}

int main(int argc,char ** argv)
{
   glutInit(&argc,argv);
   glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB );
   glutInitWindowPosition(0,0);
   glutInitWindowSize(200,200);
   glutCreateWindow("OpenGL Uygulamaları-I");
   ayarlar();
   glutDisplayFunc(gosterim);
   glutMainLoop();
   return 0;
}
```

Programda ilk olarak bir pencere yaratılmakta daha sonra da gösterim fonksiyonu ayarlanmaktadır. Gösterim fonksiyonu içerisinde de her defasında çizilecek olan grafik çizilmektedir. Bu hali ile verilen kod basit bir openGL programının iskeletini oluşturmaktadır. Program çalıştırıldığında elde edilen ekran görüntüsü Şekil 1'de verilmiştir.



Şekil 1. OpenGL ile çizilmiş basit bir şekil

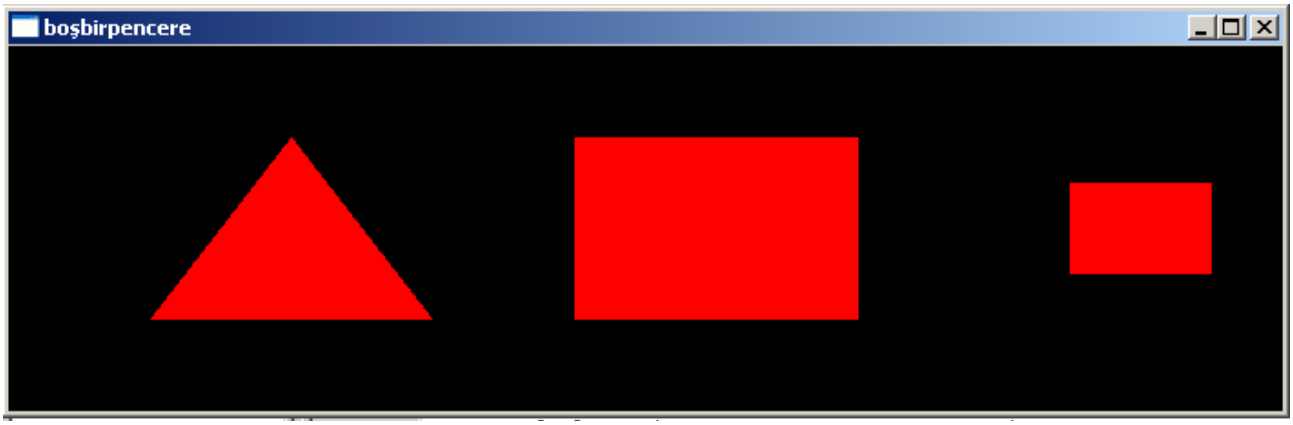
2.5.OpenGL ile Birden Fazla Şekil Çizimi

Birden fazla şekil çizmek için yukarıda verilen iskelet program üzerinde sadece gosterim isimli fonksiyon aşağıdaki şekilde değiştirilir.

,

```
void gosterim(void)
{
    glLoadIdentity();
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(1.0, 0.0, 0.0);
    glOrtho(-2.0, 7.0, -2.0, 2.0, -1.0, 1.0);
    glBegin(GL_TRIANGLES);
            glVertex3f( 0.0f, 1.0f, 0.0f);
            glVertex3f(-1.0f,-1.0f, 0.0f);
            glVertex3f( 1.0f,-1.0f, 0.0f);
    glEnd();

    glTranslatef(3.0f,0.0f,0.0f);
    glBegin(GL_QUADS);
            glVertex3f(-1.0f, 1.0f, 0.0f);
            glVertex3f( 1.0f, 1.0f, 0.0f);
            glVertex3f( 1.0f,-1.0f, 0.0f);
            glVertex3f(-1.0f,-1.0f, 0.0f);
   glEnd();

    glTranslatef(3.0f,0.0f,0.0f);
    glBegin(GL_POLYGON);
       glVertex2f(-0.5, -0.5);
       glVertex2f(-0.5, 0.5);
       glVertex2f(0.5, 0.5);
       glVertex2f(0.5, -0.5);
    glEnd();
    glFlush();

}
```

Verilen programın ekran görüntüsü Şekil 2'de verilmiştir.



Şekil 2. OpenGL ile birden çok şekil çizmek

Tartışma Sorusu-1 : Gösterim fonksiyonunda kullanılan *glLoadIdentity();* fonksiyonunun işlevi nedir? Neden kullanılmıştır? Bu fonksiyon kullanılmasaydı ekran görüntüsü nasıl olurdu? Tartışınız…

,

2.6.OpenGL ile 3B Şekil Çizimi

Verilen taslak program üzerinde main, gösterim fonksiyonu aşağıdaki gibi değiştirilerek 3B şekiller çizilebilmektedir. Yazılan yeni openGL kodu aşağıda verilmiştir.

```
#include "GL/glut.h"
#include <stdlib.h>

GLfloat		rtri;
GLfloat		rquad;

void ayarlar(void)
{
    glClearColor(0.0,0.0,0.0,0.0);
    glMatrixMode(GL_PROJECTION);
    glOrtho(-7.0, 7.0, -2.0, 2.0, -5.0, 5.0);
    rtri = 0.0f;
}

void idle()
{
   rtri +=0.2f;
   rquad+=0.15f;
  glutPostRedisplay();
}

void gosterim(void)
{
  glEnable(GL_DEPTH_TEST);
  glColor3f(1.0, 0.0, 0.0);
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
  glPushMatrix();
            glTranslatef(-1.5f,0.0f,0.0f);
            glRotatef(rtri,0.0f,1.0f,0.0f);

            glBegin(GL_TRIANGLES);
              glColor3f(1.0f,0.0f,0.0f);
              glVertex3f( 0.0f, 1.0f, 0.0f);
              glColor3f(0.0f,1.0f,0.0f);
               glVertex3f(-1.0f,-1.0f, 1.0f);
               glColor3f(0.0f,0.0f,1.0f);
               glVertex3f( 1.0f,-1.0f, 1.0f);

               glColor3f(1.0f,0.0f,0.0f);
              glVertex3f( 0.0f, 1.0f, 0.0f);
              glColor3f(0.0f,0.0f,1.0f);
              glVertex3f( 1.0f,-1.0f, 1.0f);
              glColor3f(0.0f,1.0f,0.0f);
              glVertex3f( 1.0f,-1.0f, -1.0f);

             glColor3f(1.0f,0.0f,0.0f);
            glVertex3f( 0.0f, 1.0f, 0.0f);
            glColor3f(0.0f,1.0f,0.0f);
            glVertex3f( 1.0f,-1.0f, -1.0f);
           glColor3f(0.0f,0.0f,1.0f);
```

```
            glVertex3f(-1.0f,-1.0f, -1.0f);

            glColor3f(1.0f,0.0f,0.0f);
            glVertex3f( 0.0f, 1.0f, 0.0f);
            glColor3f(0.0f,0.0f,1.0f);
            glVertex3f(-1.0f,-1.0f,-1.0f);
            glColor3f(0.0f,1.0f,0.0f);
            glVertex3f(-1.0f,-1.0f, 1.0f);
          glEnd();
  glPopMatrix();

  glTranslatef(4.0f,0.0f,0.0f);
  glPushMatrix();
            glRotatef(rtri,1.0f,1.0f,1.0f);

            glBegin(GL_QUADS);
              glColor3f(0.0f,1.0f,0.0f);
              glVertex3f( 1.0f, 1.0f,-1.0f);
              glVertex3f(-1.0f, 1.0f,-1.0f);
              glVertex3f(-1.0f, 1.0f, 1.0f);
              glVertex3f( 1.0f, 1.0f, 1.0f);

              glColor3f(1.0f,0.5f,0.0f);
              glVertex3f( 1.0f,-1.0f, 1.0f);
              glVertex3f(-1.0f,-1.0f, 1.0f);
              glVertex3f(-1.0f,-1.0f,-1.0f);
              glVertex3f( 1.0f,-1.0f,-1.0f);

              glColor3f(1.0f,0.0f,0.0f);
              glVertex3f( 1.0f, 1.0f, 1.0f);
              glVertex3f(-1.0f, 1.0f, 1.0f);
              glVertex3f(-1.0f,-1.0f, 1.0f);
              glVertex3f( 1.0f,-1.0f, 1.0f);

              glColor3f(1.0f,1.0f,0.0f);
              glVertex3f( 1.0f,-1.0f,-1.0f);
              glVertex3f(-1.0f,-1.0f,-1.0f);
              glVertex3f(-1.0f, 1.0f,-1.0f);
              glVertex3f( 1.0f, 1.0f,-1.0f);

              glColor3f(0.0f,0.0f,1.0f);
              glVertex3f(-1.0f, 1.0f, 1.0f);
              glVertex3f(-1.0f, 1.0f,-1.0f);
              glVertex3f(-1.0f,-1.0f,-1.0f);
              glVertex3f(-1.0f,-1.0f, 1.0f);

              glColor3f(1.0f,0.0f,1.0f);
              glVertex3f( 1.0f, 1.0f,-1.0f);
              glVertex3f( 1.0f, 1.0f, 1.0f);
              glVertex3f( 1.0f,-1.0f, 1.0f);
              glVertex3f( 1.0f,-1.0f,-1.0f);
            glEnd();
  glPopMatrix();
  glutSwapBuffers();
}

int main(int argc,char ** argv)
```

```
{
   glutInit(&argc,argv);
   glutInitDisplayMode(GLUT_DEPTH | GLUT_DOUBLE | GLUT_RGBA);
   glutInitWindowPosition(0,0);
   glutInitWindowSize(900,400);
   glutCreateWindow("pencere");
   ayarlar();
   glutDisplayFunc(gosterim);
   glutIdleFunc(idle);
   glutMainLoop();
   return 0;
}
```

Tartışma Sorusu-2 : rtri ve **rquad** değişkenleri ne için kullanılmıştır. Bu değişkenlerin kullanımına alternatif ne gibi yöntemler önerilebilir? Tartışınız.

Tartışma Sorusu-3 : glRotatef() ve glTransletef fonksiyonlarının kullanım amaçlarını söyleyiniz.

Tartışma Sorusu-4 : glRotatef() ve glTransletef fonksiyonlarının kullanım sırasını yer değiştirdiğimizde ne gibi sonuçlar ortaya çıkar. Tartışınız…

Tartışma Sorusu-5 : Daire, silindir, daire halkası gibi geometrik şekiller çizmek için kullanılabilecek yöntemleri tartışınız…

3. Deneye Hazırlık

Bu bölüm, deneye gelmeden önce her öğrenci tarafından yapılması gereken maddeleri içermektedir.

1. Deneye gelmeden önce Dev C++ programı bilgisayara kurulmalı ve temel düzeyde programın kullanılışı öğrenilmelidir. Programı şu adresten (http://www.bloodshed.net/download.html) indirilebilir.
2. Ek-1'de verilen adımlar takip edilerek Dev C++ programına OpenGL ve GLUT kütüphaneleri kurulmalıdır.
3. Deney föyü dikkatlice okunmalı ve deneye hazırlık soruları cevaplanmalıdır. Deney uygulama yönergesinde gerekli açıklamalar bulunmaktadır. Deney uygulama yönergesi web sayfasında mevcuttur.
4. http://ceng.ktu.edu.tr/labs/glut2_2011.zip adresinde verilen örnek program yazılarak koşulmalı ve programın çalışma mantığı anlaşılmalıdır.

4. Deneyin Yapılışı

1. OpenGL hakkında temel bilgiler soru-cevap şeklinde sorgulanır.
2. GLUT kütüphanesi nedir ve ne için kullanılır sorularına cevap verilir.
3. GLUT kütüphanesi kullanılarak tasarlanan iskelet openGL programı yazılır.
4. Basit bir kare şekli çizen program incelenir.
5. OpenGL kullanarak birden fazla 2 boyutlu geometrik şekil çizimi gerçekleştiren C++ programı incelenir ve gerekli sorular cevaplanır.
6. 3 boyutlu şekil çizimi gerçekleştirilen C++ programı incelenir ve gerekli tartışma soruları cevaplanır.

,

7. Daire çizimi gerçekleştiren openGL programı öğrenciler tarafından tasarlanır. Kullanılan teknikler tartışılır.
8. Daire halkası çizimi gerçekleştiren openGL programı öğrenciler tarafından tasarlanır. Kullanılan teknikler tartışılır.
9. Silindir çizimi gerçekleştiren openGL programı öğrenciler tarafından tasarlanır. Kullanılan teknikler tartışılır.
10. Silindir halkası çizimi gerçekleştiren openGL programı öğrenciler tarafından tasarlanır. Kullanılan teknikler tartışılır.

5. Deney Soruları

1. OpenGL nedir? Ne için kullanılır?
2. OpenGL gibi bir grafik kütüphanesi kullanmanın avantajları nelerdir?
3. GLUT nedir? Ne için kullanılır?
4. C++'da yazılmış iskelet OpenGL programını açıklayınız
5. 2 boyutlu kare şekli çizen openGL komutlarını açıklayınız.
6. 3 boyutlu küp çizen openGL komutlarını açıklayınız.
7. OpenGL kullanarak nasıl daire çizileceğini açıklayınız.
8. OpenGL kullanarak nasıl elips çizileceğini açıklayınız.

6. Kaynakça

1. Addison Wesley, "OpenGL Programming Guide", 6th Edition, 2008.
2. http://www.opengl.org
3. http://www.lighthouse3d.com/opengl/glut/
4. http://nehe.gamedev.net/

,

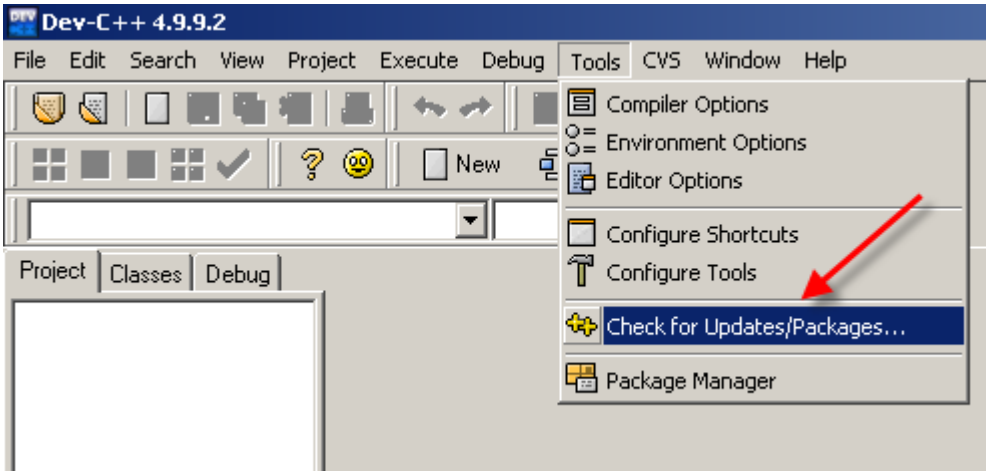
7. Ek 1

7.1.Adım 1

Dev C++ programını kurun

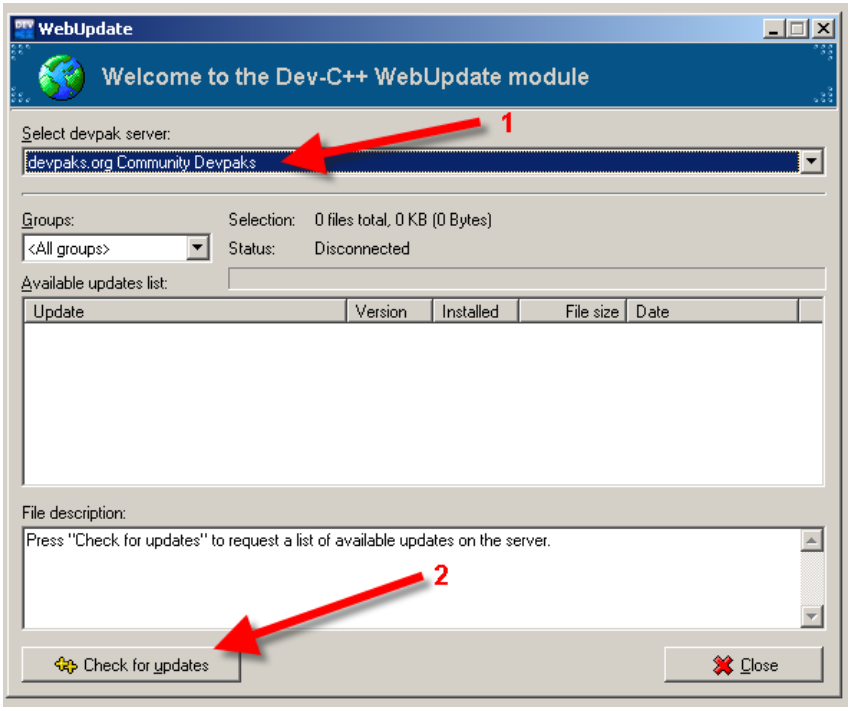
7.2.Adım 2

Tools / Check for Updates/Packages…

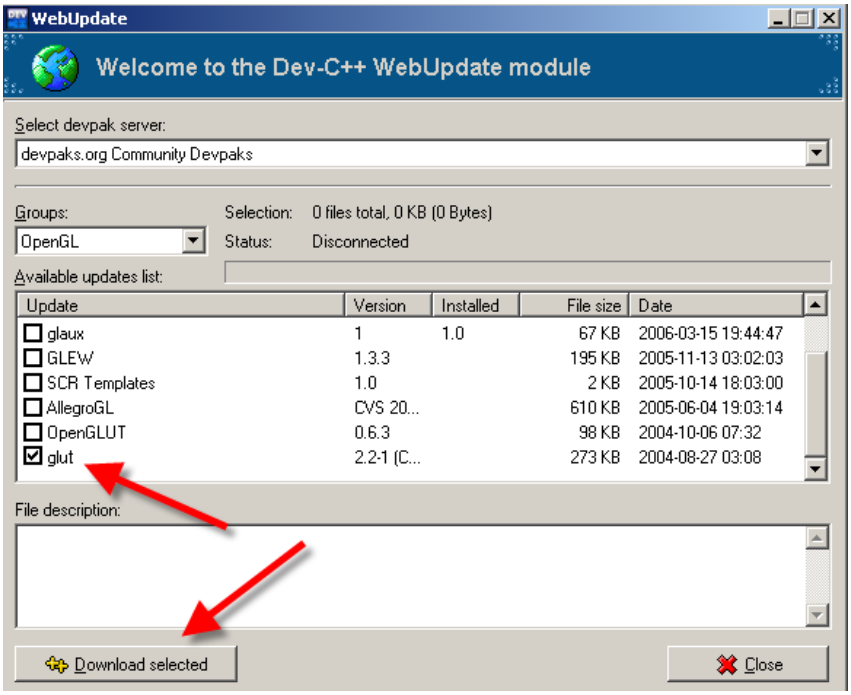


,

7.3.Adım 3



7.4.Adım 4



,

Ek-2