كتاب رقم (1A) من سلسلة برمجة الرسم بلغة VC PLUS PLUS باستخدام OpenGL

هذه النسخة بتاريخ: 2007/9/29

مبادئ برمجة الرسم بلغة

Microsoft VC PLUS PLUS 7.0

باستخدام OpenGL

في هذا الكتاب يتم استخدام مكتبة GLUT.H
لربط النوافذ بمكتبة OpenGL

برمجة: البراء عبد الرؤوف الرملي

طرابلس / ليبيا

هذا الكتاب مجاني

مقدمة

في هذا الكتاب قمت بجمع تلخيص وشرح لدوال مكتبة الرسم OpenGL, وقد استخدمت دوال GLUT لربط النوافذ بمكتبة OpenGL , وحاولت أن يكون بسيط في أسلوبه , أرجو الله أن ينفع به.

وقد أسميته:

مبادئ برمجة الرسم بلغة Microsoft VC PLUS PLUS 7.0 باستخدام OpenGL

وللعلم فقد استفدت كثيرا من موقع "الفريق العربي للبرمجة":
www.arabteam2000-forum.com

وأريد أن أنبه على أن الكتب العربية لازالت تحتاج إلى تطوير وإضافات , وهذا يقع على عاتقنا جميعا حتى تعم الفائدة , لأنه ما لم نتشارك بأفكارنا , فلن نتقدم خطوة إلى الأمام.

لأي تعليق أو ملاحظة على الكتاب "أرحب بالنقد البناء":

sbr_system@yahoo.com

كما يمكنك زيارة موقعي:

www.SBRsystem.8m.com

البراء عبد الرؤوف الرملي

طرابلس/ليبيا

2007/9/29

الفهرس

عن مكتبة OpenGL

قابلة للحمل تنفذ على أنظمة مختلفة مثل:

- **Microsoft Windows**
- **Linux**
- **UNIX-based systems**
- **Mac OS X**
- **game consoles by Nintendo and Sony such as the PlayStation 3**

مجال الاستخدام هو الرسوميات الاحترافية مثل:

- **ArchiCAD**
- **Autodesk (AutoCAD 2000)**
- **Quake (GL-Quake, Quake II & III)**
- **Microstation 95**
- **MathGL3D (Mathematica)**
- **Maya (Character animation, Modeling)**

الحقوق والسماحيات: مفتوحة المصادر Open Source.

ما الذي يلزم لاستخدام مكتبة GLUT.H

سنعتمد في هذا الكتاب على Visual C++ 7.0
في بيئة Microsoft Visual Studio .NET 2003

قم بنسخ الملفات التالية إلى المسارات المبينة (الملفات الثلاثة مرفقة مع الكتاب):

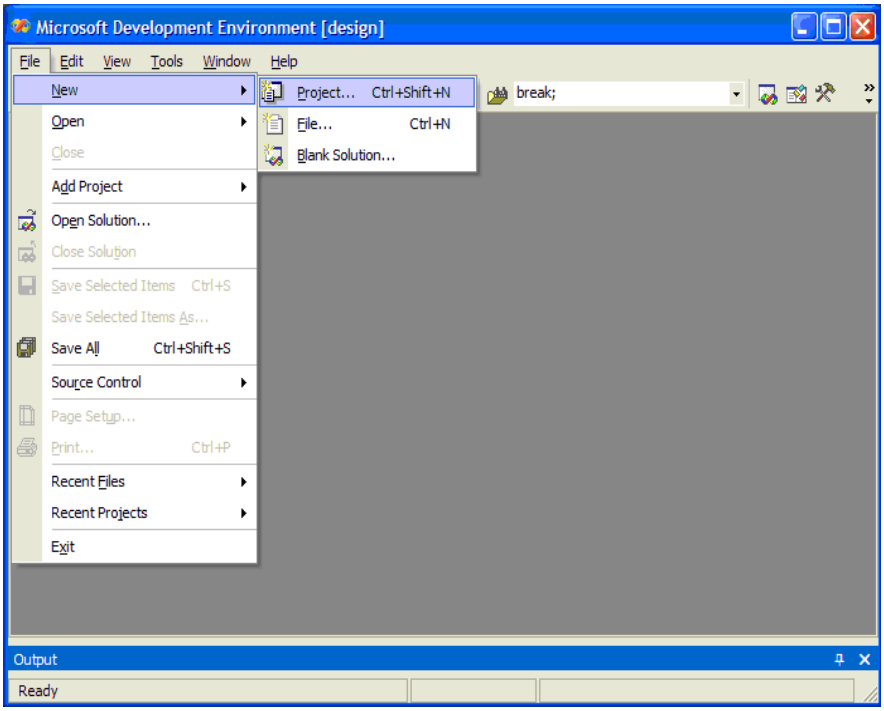
C:\Program Files\Microsoft Visual Studio .NET2003\Vc7\PlatformSDK\Lib
glut32.lib

C:\Program Files\Microsoft Visual Studio .NET2003\Vc7\PlatformSDK\Include\gl
glut.h

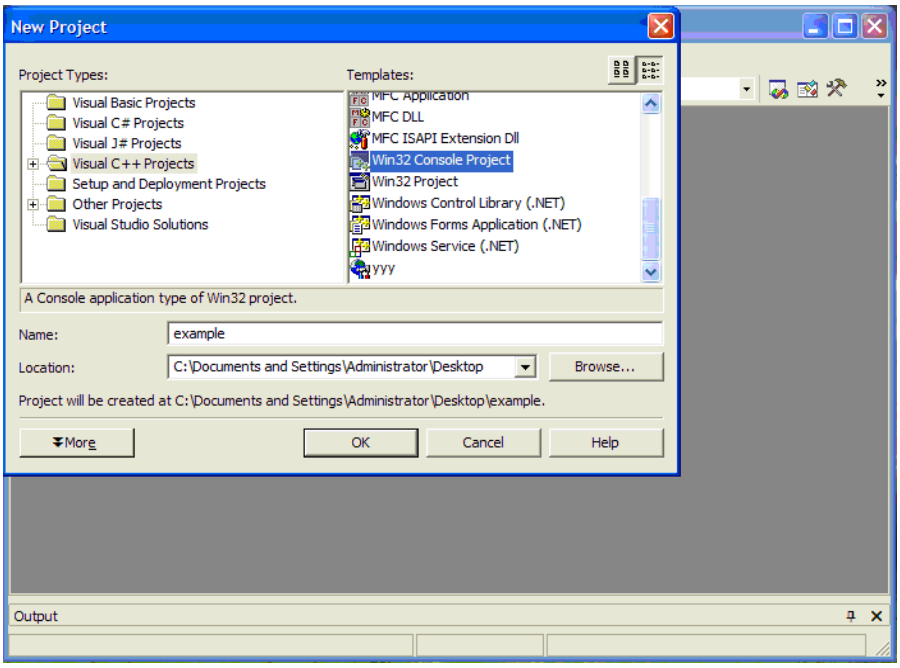
c:\windows\system32
glut32.dll

فتح مشروع جديد في Visual C PLUS PLUS

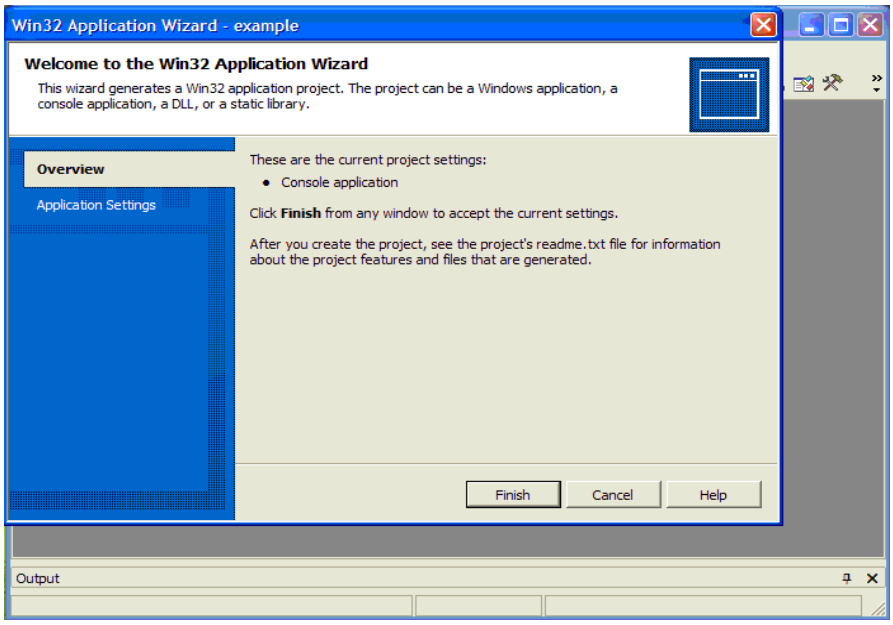
قم بفتح Microsoft Visual Studio .NET 2003 ثم اضغط على File ثم New ثم Project



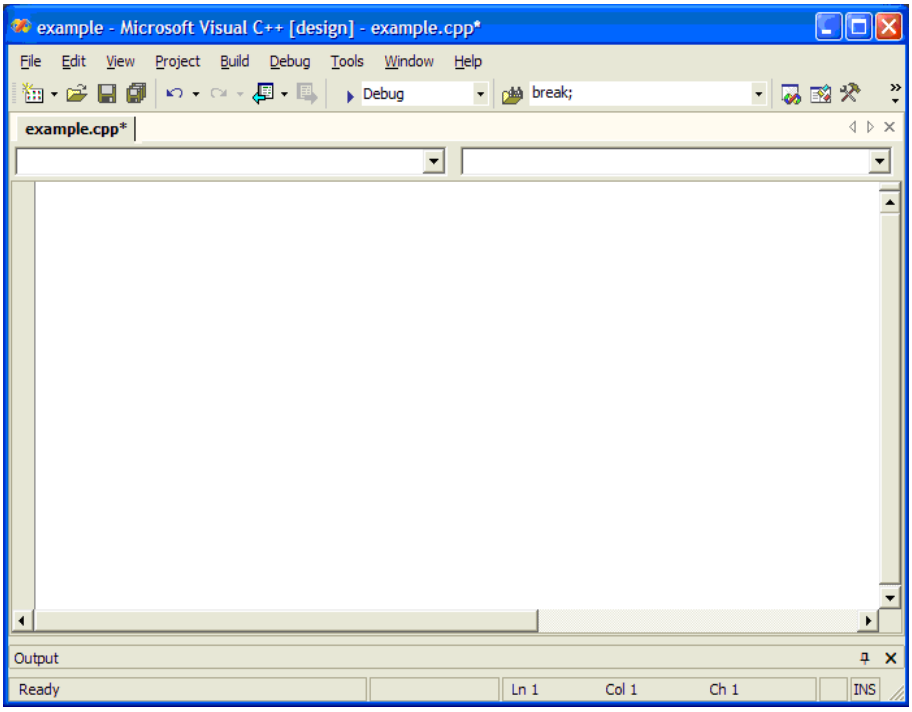
فتظهر لك النافذة السفلية , اختر Visual C++ ثم أكتب اسما للمشروع ثم اضغط على Win32 Console Project ثم OK



فتظهر لك النافذة السفلية فاضغط **Finish**



فتظهر لك صفحة المشروع , امسح النص الموجود بها لتكتب برنامجك



الشكل العام لبرامج الرسم

`#include "stdafx.h"`	افتراضية من Microsoft VC PLUS PLUS
`#include <GL/glut.h>`	استدعاء مكتبة GLUT.H

`void ini_graph()`
`{`
`glClearColor(0.3,0.8,1.0,0.0);`
`glColor3f(1.0,1.0,1.0);`
`glPointSize(1.0);`
`glMatrixMode(GL_PROJECTION);`
`glLoadIdentity();`
`gluOrtho2D(0 , 700 , 0 , 600);`
`}`
	لإعداد بيئة الرسم

`void ini_wind()`
`{`
`glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);`
`glutInitWindowSize(700,600);`
`glutInitWindowPosition(0,0);`
`glutCreateWindow("");`
`}`
	لإعداد نافذة الرسم

`void graph()`
`{`
`glClear(GL_COLOR_BUFFER_BIT);`
دوال الرسم تكتب هنا
`glFlush();`
`}`
	دالة بيانات الرسم

`int main()` `{`	
`ini_graph();`	استدعاء دالة لإعداد بيئة الرسم
`ini_wind();`	استدعاء دالة لإعداد نافذة الرسم
`glutDisplayFunc(graph);`	استدعاء دالة لعرض دالة بيانات الرسم
`glutMainLoop();`	لتثبيت الرسم , حيث يتم تنفيذ البرنامج بشكل متواصل
`return 0;` `}`	

أولا:
شرح لدوال الرسم
في مكتبة GLUT.H

لرسم الأشكال الأساسية

أولا يجب أن تعلم أن نافذة الرسم قسمة إلى 4 أقسام بالنسبة للدالة (glVertex2f) وأخواتها بحيث تكون نقطة الأصل في مركز النافذة كما يظهر في الشكل التالي:

لرسم الأشكال الأساسية (النقطة-خط مستقيم-مثلث-مستطيل-وغيرها) يلزم 3 دوال:

- دالة لتحديد نوع الشكل المراد رسمه (عدد النقاط)
- دوال لتحديد نقاط الشكل
- دالة للدلالة على انتهاء النقاط

دالة لتحديد نوع الشكل المراد رسمه (عدد النقاط)

glBegin(NAME);

NAME	الوصف
GL_LINES	لرسم خط .. ولهذا يلزمك نقطتين
GL_LINE_STRIP	رسم خطوط متصلة
GL_LINE_LOOP	نفس الأمر السابق لكن يتم وصل آخر نقطة بأول نقطة ليتم رسم شكل مغلق
GL_TRIANGLES	لرسم مثلث .. وتحتاج إلى ثلاث نقاط أو مضاعفات الثلاثة
GL_TRIANGLE_STRIP	لرسم مثلثات متصلة
GL_TRIANGLE_FAN	لرسم مثلثات تتصل بنقطة المنتصف (غالبا تستخدم لرسم الإشكال الدائرية)
GL_QUADS	لرسم إشكال مربعة أو مستطيلة (يعني إشكال ذات أربع رؤوس) وتحتاج إلى أربع نقاط لرسم شكل رباعي واحد .. وثمان نقاط لرسم شكلين رباعيين
GL_QUAD_STRIP	لرسم أشكال رباعية متصلة
GL_POLYGON	لرسم مضلع .

دالة لتحرير نقطة ثنائية البعد

glVertex2f(x,y);

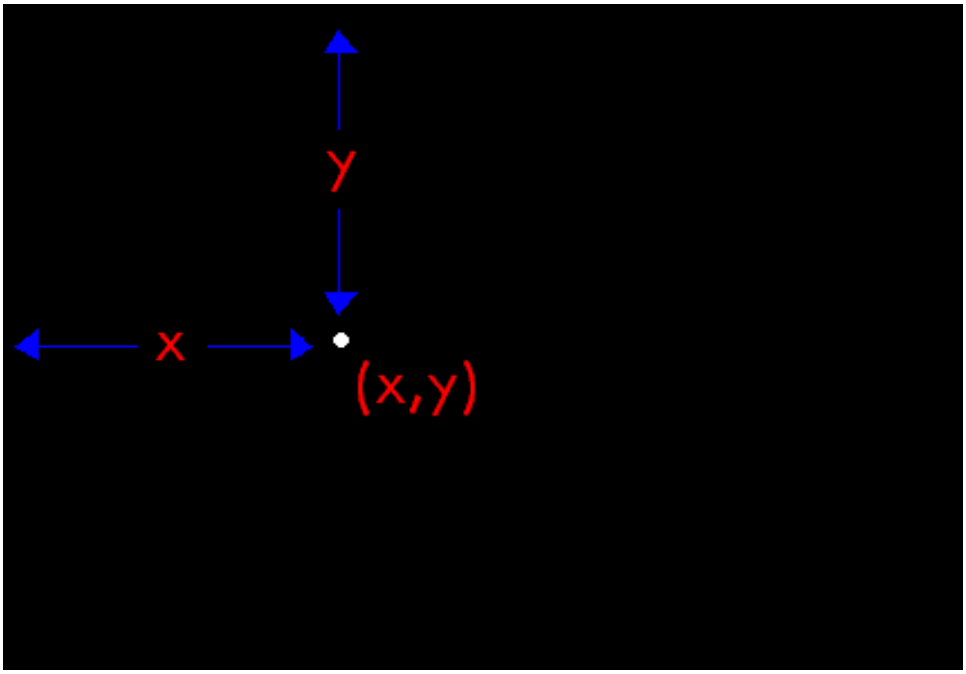
أما لتحرير نقطة ثلاثية البعد

glVertex2f(x,y,z);

دالة للدلالة على انتهاء النقاط

glEnd();

لرسم نقطة



فمثلا لرسم نقطة واحدة يلزم 3 دوال:

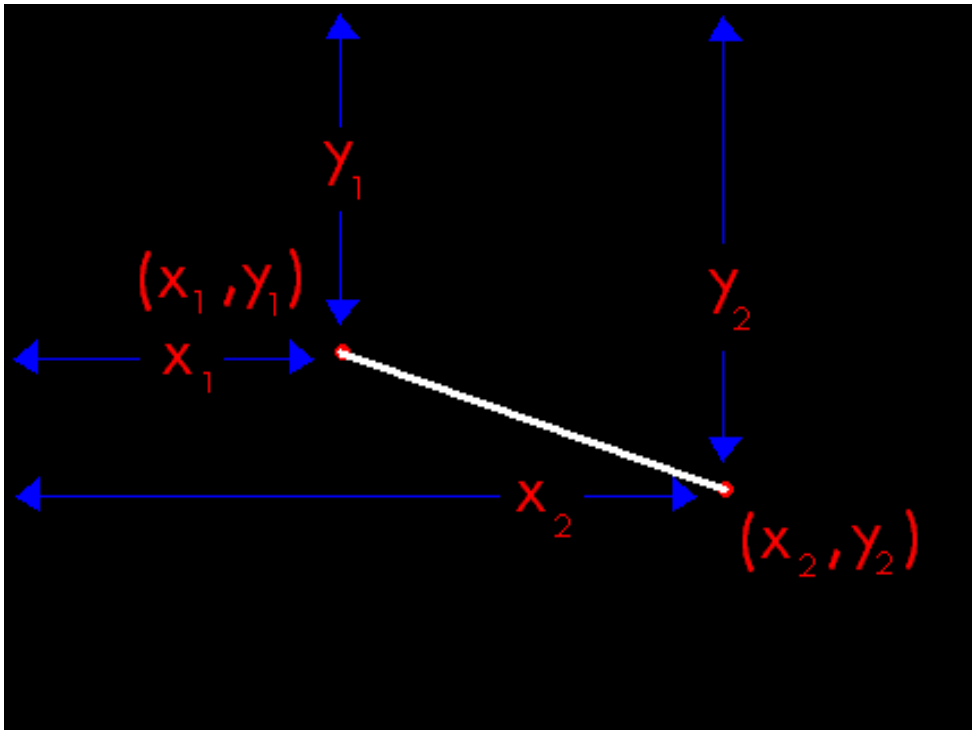
glBegin(GL_POINTS);	**تحديد نوع الشكل وفي هذه الحالة نختار نقاط منفردة**
glVertex2f(x,y);	**دالة لتحديد نقطة حيث)x,y(متغيرين صحيحين مثلان إحداثي النقطة**
glEnd();	**دالة للدلالة على انتهاء النقاط**

فقمنا بجمع الـ 3 دوال داخل دالة باسم point ومن ثم نقوم باستدعائها:

#include "stdafx.h"	افتراضية من Microsoft VC PLUS PLUS
#include <GL/glut.h>	استدعاء مكتبة glut.h
void ini_graph() **{**	لإعداد بيئة الرسم
glClearColor(0.0,0.0,0.0,0.0);	تحدد لون الخلفية
glColor3f(1.0,1.0,1.0);	تحديد لون خط الرسم
glPointSize(1.0);	تحديد حجم النقطة
glMatrixMode(GL_PROJECTION);	نظام التعامل مع الشاشة
glLoadIdentity();	نرجع لنقطة الأصل إلى منتصف النافذة
gluOrtho2D(0,700,0,600);	التعامل مع رسومات الشاشة بالنسب
}	
void ini_wind() **{**	لإعداد نافذة الرسم
glutInitDisplayMode(GLUT_SINGLE | **GLUT_RGB);**	نظام التعامل مع الألوان
glutInitWindowSize(700,600);	تحديد عرض و طول الشاشة
glutInitWindowPosition(0,0);	بتحديد المكان المراد ظهور النافذة
glutCreateWindow("رسم نقطة");	لعرض نص على الشريط العلوي للنافذة

}	
void point(float x, float y) **{** **glBegin(GL_POINTS);** **glVertex2f(x,y);** **glEnd();** **}**	دالة لرسم نقطة
void graph() **{**	دالة بيانات الرسم
`glClear(GL_COLOR_BUFFER_BIT);`	لمسح النافذة
point(0.0,0.0,1.0,1.0);	
`glFlush();` **}**	هذه الدالة هي المسئولة عن إظهار الصورة على الشاشة حيث تأمر كرت الشاشة برسم الصورة، وبدونها لن تعمل جميع لأسطر السابقة
int main() **{**	
ini_graph();	استدعاء دالة لإعداد بيئة الرسم
ini_wind();	استدعاء دالة لإعداد نافذة الرسم
glutDisplayFunc(graph);	استدعاء دالة لعرض دالة بيانات الرسم
glutMainLoop();	لتثبيت الرسم حيث يتم تنفيذ البرنامج بشكل متواصل
return 0; **}**	

لرسم خط مستقيم



لرسم خط مستقيم من الإحداثي (x1,y1) إلى الإحداثي (x2,y2) يلزم 4 دوال:

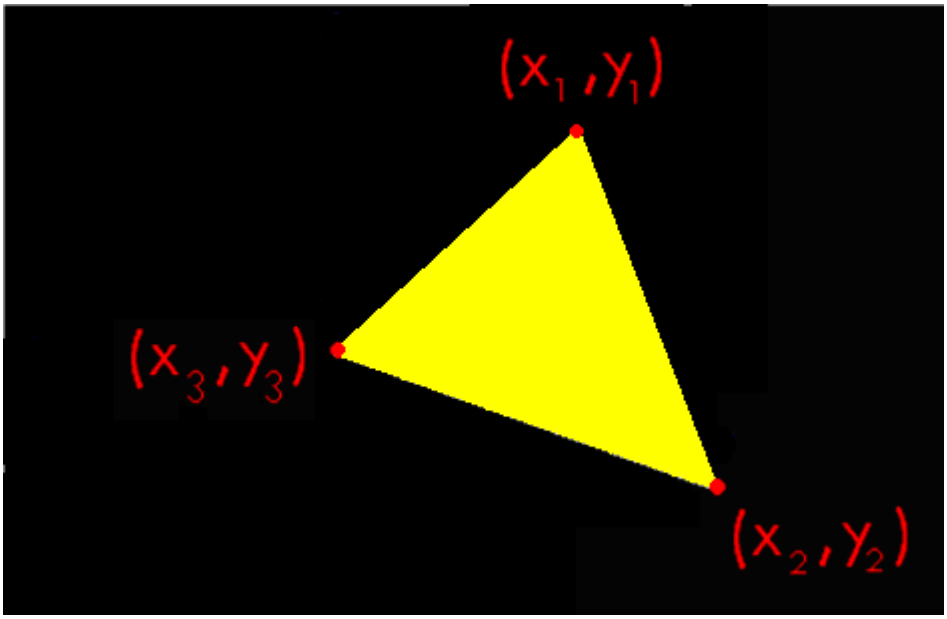
glBegin(GL_LINES);	نوع الشكل خط مستقيم
glVertex2f(x1,y1); **glVertex2f(x2,y2);**	لتحديد نقطة بداية ونهاية المستقيم
glEnd();	دالة للدلالة على انتهاء النقاط

فقمنا بجمع الـ 4 دوال داخل دالة باسم line ومن ثم نقوم باستدعائها:

#include "stdafx.h"	افتراضية من Microsoft VC PLUS PLUS
#include <GL/glut.h>	استدعاء مكتبة glut.h
void ini_graph() **{**	لإعداد بيئة الرسم
glClearColor(0.0,0.0,0.0,0.0);	تحدد لون الخلفية
glColor3f(1.0,1.0,1.0);	تحديد لون خط الرسم
glPointSize(1.0);	تحديد حجم النقطة
glMatrixMode(GL_PROJECTION);	نظام التعامل مع الشاشة
glLoadIdentity();	نرجع لنقطة الأصل وهي منتصف النافذة
gluOrtho2D(0,700,0,600);	التعامل مع رسومات الشاشة بالنسب
}	
void ini_wind() **{**	لإعداد نافذة الرسم
glutInitDisplayMode(GLUT_SINGLE | **GLUT_RGB);**	نظام التعامل مع الألوان

glutInitWindowSize(700,600);	تحديد عرض و طول الشاشة
glutInitWindowPosition(0,0);	بتحديد المكان المراد ظهور النافذة
glutCreateWindow("رسم خط مستقيم");	لعرض نص على الشريط العلوي للنافذة
}	
void line(float x1, float y1,float x2, float y2) **{** **glBegin(GL_LINES);** **glVertex2f(x1,y1);** **glVertex2f(x2,y2);** **glEnd();** **}**	دالة لرسم خط مستقيم
void graph() **{**	دالة بيانات الرسم
glClear(GL_COLOR_BUFFER_BIT);	لمسح النافذة
line(0.0,0.0,1.0,1.0);	
glFlush(); **}**	هذه الدالة هي المسئولة عن إظهار الصورة على الشاشة حيث تأمر كرت الشاشة برسم الصورة , وبدونها لن تعمل جميع لأسطر السابقة
int main() **{**	
ini_graph();	استدعاء دالة لإعداد بيئة الرسم
ini_wind();	استدعاء دالة لإعداد نافذة الرسم
glutDisplayFunc(graph);	استدعاء دالة لعرض دالة بيانات الرسم
glutMainLoop();	لتثبيت الرسم حيث يتم تنفيذ البرنامج بشكل متواصل
return 0; **}**	

لرسم مثلث وتلوينه



لرسم مثلث رؤوسه (x1,y1) (x2,y2) (x3,y3) وتلوينه يلزم 6 دوال:

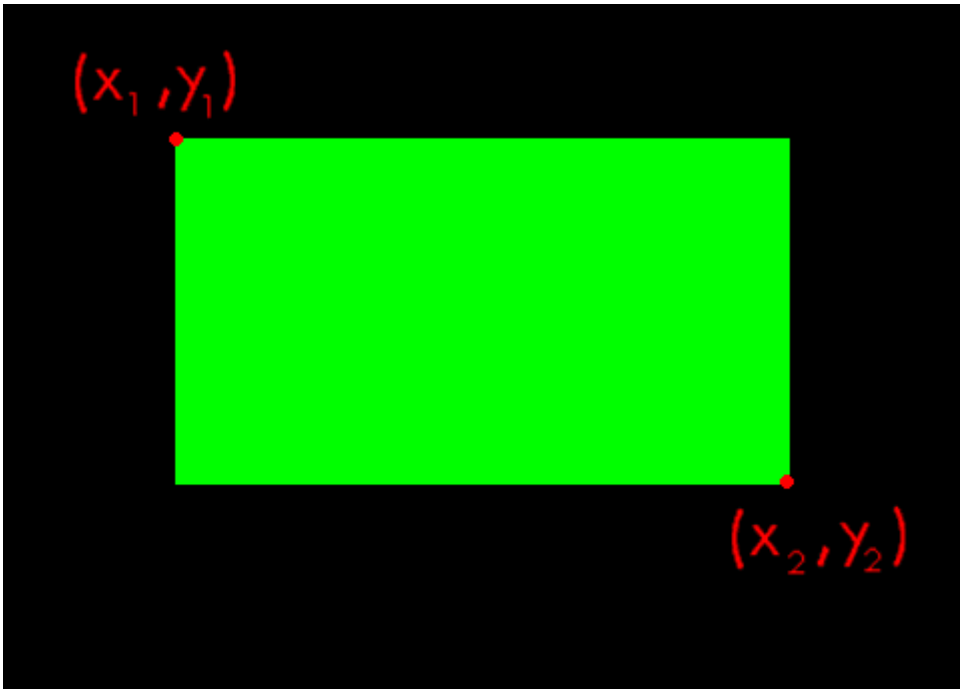
glColor3f(0.7, 0.7, 0.0);	لون المثلث وهو هنا الأصفر
glBegin(GL_TRIANGLES);	نوع الشكل مثلث
glVertex2f(x1,y1); **glVertex2f(x2,y2);** **glVertex2f(x3,y3);**	لتحديد إحداثيات رؤوس المثلث
glEnd();	دالة للدلالة على انتهاء النقاط

فقمنا بجمع الـ 5 دوال داخل دالة باسم triangle ومن ثم نقوم باستدعائها:

#include "stdafx.h"	افتراضية من Microsoft VC PLUS PLUS
#include <GL/glut.h>	استدعاء مكتبة glut.h
void ini_graph() **{**	لإعداد بيئة الرسم
glClearColor(0.0,0.0,0.0,0.0);	تحدد لون الخلفية
glColor3f(1.0,1.0,1.0);	تحديد لون خط الرسم
glPointSize(1.0);	تحديد حجم النقطة
glMatrixMode(GL_PROJECTION);	نظام التعامل مع الشاشة
glLoadIdentity();	نرجع لنقطة الأصل وهي منتصف النافذة
gluOrtho2D(0,700,0,600);	التعامل مع رسومات الشاشة بالنسب
}	
void ini_wind() **{**	لإعداد نافذة الرسم
glutInitDisplayMode(GLUT_SINGLE | **GLUT_RGB);**	نظام التعامل مع الألوان
glutInitWindowSize(700,600);	تحديد عرض و طول الشاشة

glutInitWindowPosition(0,0);	بتحديد المكان المراد ظهور النافذة
glutCreateWindow("رسم مثلث وتلوينه");	لعرض نص على الشريط العلوي للنافذة
}	
void triangle(float x1, float y1,float x2, float y2,float x3, float y3) **{** glColor3f(0.7, 0.7, 0.0); **glBegin(GL_TRIANGLES);** **glVertex2f(x1,y1);** **glVertex2f(x2,y2);** **glVertex2f(x3,y3);** **glEnd();** **}**	دالة لرسم مثلث وتلوينه
void graph() **{**	دالة بيانات الرسم
glClear(GL_COLOR_BUFFER_BIT);	لمسح النافذة
triangle**(-0.5,0.5,0.5,0.5,0.0,-0.5);**	
glFlush(); **}**	هذه الدالة هي المسئولة عن إظهار الصورة على الشاشة حيث تأمر كرت الشاشة برسم الصورة , وبدونها لن تعمل جميع الأسطر السابقة
int main() **{**	
ini_graph();	استدعاء دالة لإعداد بيئة الرسم
ini_wind();	استدعاء دالة لإعداد نافذة الرسم
glutDisplayFunc(graph);	استدعاء دالة لعرض دالة بيانات الرسم
glutMainLoop();	لتثبيت الرسم حيث يتم تنفيذ البرنامج بشكل متواصل
return 0; **}**	

لرسم مستطيل وتلوينه



لرسم مستطيل ركنيه (x1,y1) (x2,y2) وتلوينه يلزم 6 دوال:

glColor3f(0.0, 1.0, 0.0);	لون المستطيل وهو هنا الأخضر
glBegin(GL_TRIANGLES);	نوع الشكل مثلث
glVertex2f(x1,y1); **glVertex2f(x2,y1);** **glVertex2f(x2,y2);** **glVertex2f(x1,y2);**	لتحديد إحداثيات أركان المستطيل
glEnd();	دالة للدلالة على انتهاء النقاط

فقمنا بجمع الـ 5 دوال داخل دالة باسم rectangle ومن ثم نقوم باستدعائها:

#include "stdafx.h"	افتراضية من Microsoft VC PLUS PLUS
#include <GL/glut.h>	استدعاء مكتبة glut.h
void ini_graph() **{**	لإعداد بيئة الرسم
glClearColor(0.0,0.0,0.0,0.0);	تحدد لون الخلفية
glColor3f(1.0,1.0,1.0);	تحديد لون خط الرسم
glPointSize(1.0);	تحديد حجم النقطة
glMatrixMode(GL_PROJECTION);	نظام التعامل مع الشاشة
glLoadIdentity();	نرجع لنقطة الأصل وهي منتصف النافذة
gluOrtho2D(0,700,0,600);	التعامل مع رسومات الشاشة بالنسب
}	
void ini_wind() **{**	لإعداد نافذة الرسم
glutInitDisplayMode(GLUT_SINGLE | **GLUT_RGB);**	نظام التعامل مع الألوان

glutInitWindowSize(700,600);	تحديد عرض و طول الشاشة
glutInitWindowPosition(0,0);	بتحديد المكان المراد ظهور النافذة
glutCreateWindow("رسم مستطيل وتلوينه");	لعرض نص على الشريط العلوي للنافذة
}	
void rectangle(float x1, float y1,float x2,float y2) **{** **glColor3f(0.0, 1.0, 0.0);** **glBegin(GL_QUADS);** **glVertex2f(x1,y1);** **glVertex2f(x2,y1);** **glVertex2f(x2,y2);** **glVertex2f(x1,y2);** **glEnd();** **}**	دالة لرسم مستطيل وتلوينه
void graph() **{**	دالة بيانات الرسم
glClear(GL_COLOR_BUFFER_BIT);	لمسح النافذة
rectangle(-0.5,0.5,0.5, 0.0);	
glFlush(); **}**	هذه الدالة هي المسئولة عن إظهار الصورة على الشاشة حيث تأمر كرت الشاشة برسم الصورة , وبدونها لن تعمل جميع الأسطر السابقة
int main() **{**	
ini_graph();	استدعاء دالة لإعداد بيئة الرسم
ini_wind();	استدعاء دالة لإعداد نافذة الرسم
glutDisplayFunc(graph);	استدعاء دالة لعرض دالة بيانات الرسم
glutMainLoop();	لتثبيت الرسم حيث يتم تنفيذ البرنامج بشكل متواصل
return 0; **}**	

أولا:
شرح لدوال إعداد بيئة الرسم

بداية يجب أن نعرف أين يتم إظهار الرسم ولمعرفة ذلك:

يجب أن نعرف ما هو Back Buffer:

هو سطح خفي نرسم عليه ثم بعد أن ننتهي من الرسم نقوم بإحضاره إلى الشاشة , في حوال Windows عندما نرسم شي ونحركه لا تظهر الحركة ناعمة إنما يحدث وميض مزعج مشوه للرسم عند تحريكه .. السبب في ذلك أن Windows به سطح واحد للرسم فقط .

نوع الـ Buffer		واسمه البرمجي (عبارة عن قيمة)	الوصف
اللون	Color	GL_COLOR_BUFFER_BIT	تخزين ألوان البكسلات الضوئية
العمق	Depth	GL_DEPTH_BUFFER_BIT	يدعى أحياناً Z buffer ويستخدم لتخزين قيم العمق لكل بكسل والتي تمثل قياس المسافة عن عين الناظر
الحاجب	Stencil	GL_STENCIL_BUFFER_BIT	يستخدم لقصر الرسم على مناطق محددة من الشاشة .
التراكمي	Accumulation	GL_ACCUM_BUFFER_BIT	يستخدم لمراكمة مجموعة صور لتشكل صورة نهائية مركبة .

دالة تحديد لون مسح الخلفية

glClearColor(A ,B ,C ,D);

بإمكانك تحديد لون المسح لمرة واحدة فقط في بداية تطبيقك , ويمرر لهذه الدالة 4 قيم لأننا نستخدم نظام الألوان RGBA وفي هذا النظام يكون اللون الناتج خليط لـ 3 ألوان هم الأحمر والأخضر والأزرق وتكون القيم الممررة للدالة كالتالي:

عنصر	عدد عشري	يمثل درجة
A	من 0.0 إلى 1.0	اللون الأحمر
B	من 0.0 إلى 1.0	اللون الأخضر
C	من 0.0 إلى 1.0	اللون الأزرق
D	من 0.0 إلى 1.0 ويسمى معامل alpha	شفافية اللون الناتج فإذا كانت القيمة 0.0 فاللون يكون شفاف وإذا كانت 1.0 يكون غير شفاف ولكننا لم نستخدمه هنا.

كلما اقتربنا من 0.0 يكون اللون داكن وكلما اقتربنا من 1.0 يكون اللون فاتح

دالة لمسح النافذة

قبل القيام بذلك لابد من تحديد لون لمسح النافذة باستدعاء الدالة السابقة

glClear(GL_COLOR_BUFFER_BIT);

يشير المعامل GL_COLOR_BUFFER_BIT إلى نوع الـ Buffer الذي يجب مسحه وهي هنا (Buffer اللون).

كما يمكن مسح العمق (عند الرسم ثلاثي الأبعاد) واللون معا كالتالي:

```
glClear(GL_COLOR_BUFFER_BIT |
;(GL_DEPTH_BUFFER_BIT
```

دالة لتجهيز لون لمسح العمق

```
glClearDepth(Z);
```

Z عدد عشري يمثل القيمة التي يجب أن تعين لكل بكسل في Depth Buffer

لتغيير لون خط الرسم

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

هذه الدالة لتحديد لون الجسم المرسوم ويمرر لها 3 قيم تمثل درجات الألوان (الأحمر والأخضر والأزرق).

عنصر	عدد عشري	يمثل درجة
A	من 0.0 إلى 1.0	اللون الأحمر
B	من 0.0 إلى 1.0	اللون الأخضر
C	من 0.0 إلى 1.0	اللون الأزرق
D	من 0.0 إلى 1.0 ويسمى معامل alpha	شفافية اللون الناتج فإذا كانت القيمة 0.0 فاللون يكون شفاف وإذا كانت 1.0 يكون غير شفاف ولكننا لم نستخدمه هنا.

كلما اقتربنا من 0.0 يكون اللون داكن وكلما اقتربنا من 1.0 يكون اللون فاتح

ملاحظة: تكتب قبل دالة تحديد الشكل glBegin

لتغيير حجم النقطة

```
glPointSize(Z);
```

هذه الدالة لتحديد حجم النقطة حيث أن المتغير الصحيح Z يمثل الحجم فإذا كان Z=2 فإنه سيتم تمثيل النقطة الواحدة بنقطتين

ملاحظة: تكتب قبل دالة تحديد الشكل glBegin

نظام التعامل مع الشاشة

```
glMatrixMode(GL_PROJECTION);
```

ترجع إلى نقطة الأصل إلى منتصف الشاشة

glLoadIdentity();

تقوم بتصفير المصفوفة , يعني تعيد المحاور الرئيسية بحيث تكون نقطة الأصل في مركز الشاشة , نوع المصفوفة التي نتعامل معها الآن هي GL_MODELVIEW

التعامل مع رسومات النافذة بالنسب

gluOrtho2D(0 , 700 , 0 , 600);

استخدام ثلاثي الأبعاد

glEnable(GL_DEPTH_TEST);

المعامل GL_DEPTH_TEST يعني أننا نريد استخدام البعد z

لتغيير سمك الخط

glLineWidth(d);

حيث d متغير يمثل السمك المطلوب

ملاحظة: تكتب قبل دالة تحديد الشكل glBegin

دالة إزاحة نقطة الأصل

glTranslatef(x,y,z);

المعاملات الثلاثة x,y,z تمثل المحاور الرئيسية , والمكان الافتراضي لنقطة الأصل هو منتصف الشاشة , وأي إزاحة لها هنا فإن تأثيرها يستمر.

دالة الإزاحة تمكنك من نقل نقطة الأصل فمثلا إذا كانت القيم داخل الدالة كالتالي:

glTranslatef(-1.5f,0.0f,-6.0f);

فإن المحور X يتحرك باتجاه اليسار بمقدار 1.5 (تذكر أن نقطة الأصل الإفتراضية في منتصف الشاشة ولذلك تحركك إلى اليسار يعتبر قيمة سالبة)

المحور Y يساوي صفر أي أننا لن نقوم بإزاحته في هذا الاتجاه.

بينما يتحرك المحور Z داخل الشاشة بعمق 1.5 (العمق لا يظهر جليا إلا في الرسم ثلاثي الأبعاد) .

ملاحظة: تكتب قبل دالة تحديد الشكل glBegin

دالة لتوضيح الرسم

glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);

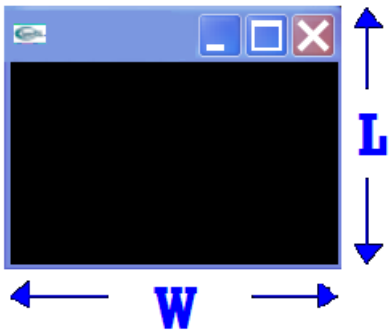
أولا:
شرح لدوال إعداد نافذة الرسم

نظام التعامل مع الألوان

glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);

تحديد طول وعرض النافذة

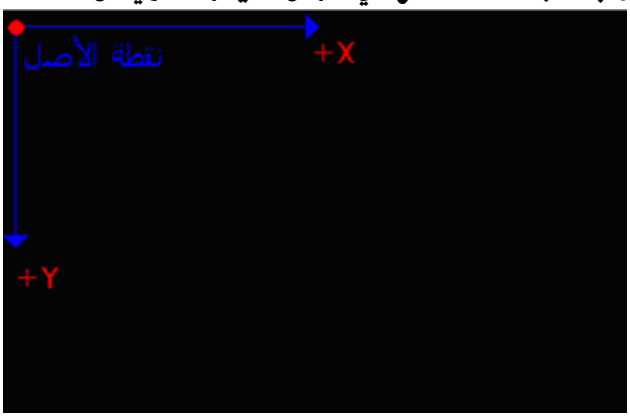
glutInitWindowSize(W,L);



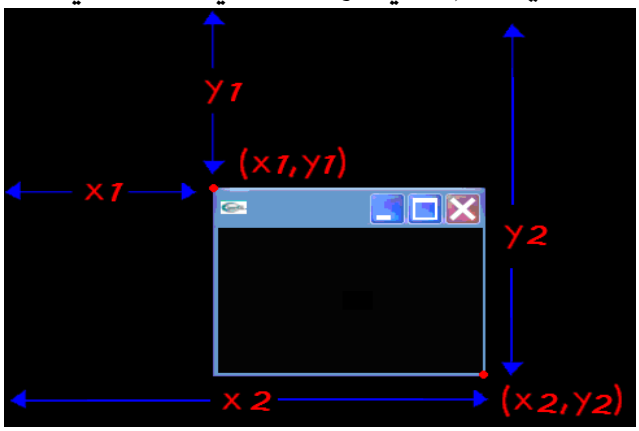
مكان ظهور النافذة

glutInitWindowPosition(0,0);

ملاحظة: (نقطة الأصل بالنسبة للشاشة تقع في الركن الأيسر العلوي من الشاشة كما يظهر في الشكل)

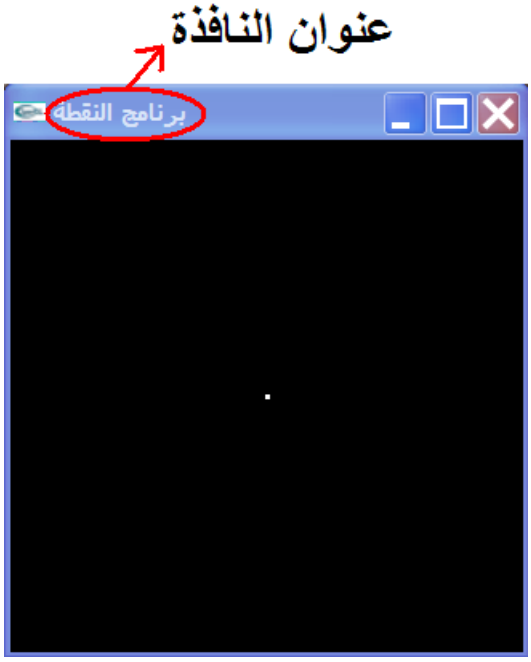


لذا يمرر الإحداثي (x,y) كما في الشكل التالي:



لعرض نص على الشريط العلوي للنافذة

```
glutCreateWindow("برنامج النقطة");
```



المراجع

المصدر	المصدر	رابط الموقع
الشمري مشرف قسم برمجة الألعاب والجرافيكس	الفريق العربي للبرمجة , قسم برمجة الألعاب والجرافكس باستخدام مكتبة OpenGL	www.arabteam2000-forum.com
CompuM4n عضو		
دروس OpenGL المترجمة تأليف علي دعيج و Basha		
كتاب إلكتروني	جامعة حلب الهندسة الإلكترونية	---