Drawing some basic primitives

Drawing some basic primitives

```
#include<Windows.h>
#include<Gl/glut.h>
void myDisplay();
void changeColor();
void reShape( int , int );
```

Drawing some basic primitives

```
int main( int argc , char** argv){
glutInit(&argc,argv);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
//glutInitWindowPosition(0,0);
//glutInitWindowSize(500,500);
glutInitWindowSize( GetSystemMetrics(SM_CXSCREEN)
,GetSystemMetrics(SM_CYSCREEN));
glutCreateWindow("Some basic primitives");
```

Drawing some basic primitives

```
// change the value of defult colo buffer
//changeColor();
glutDisplayFunc(myDisplay);
glutReshapeFunc(reShape);
glutMainLoop();
}
```

Drawing some basic primitives

```
////////////////////////////////////////
void myDisplay(){
// clear buffers
glClear( GL_COLOR_BUFFER_BIT |
GL_DEPTH_BUFFER_BIT);
// resets the current  Matrix
glLoadIdentity();
// draw our objects
glLineWidth(10.0);
```

Drawing some basic primitives

```
glBegin(GL_TRIANGLES);
glColor3b(1,1,0);
glVertex2f(0.0,5.0);
glVertex2f(4.0,-3.0);
glVertex2f(-4,-3.0);
glEnd();
// display on screen
glFlush();

}
```

Drawing some basic primitives

```
void changeColor() {
glClearColor(1.0 , 1.0 , 0.0 , 1.0);
}
```

Drawing some basic primitives

```
void reShape( int w , int h  ){
// specify View port
glViewport((GLint)0,(GLint)0,(GLsizei)w,(GLsizei)h
);
// specify projection
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(-10,10,-10,10);
glMatrixMode(GL_MODELVIEW);
}
```

Drawing some basic primitives

- `#include<Windows.h>` The header "windows.h" is needed to get Screen Size (W, H).
- `#include<Gl/glut.h>` The header "Gl/glut.h" is needed for OpenGl API.
- `int main( int argc , char** argv)` You use this to initialize your glut
- `void myDisplay();` prototype for myDisplay function that draw by objects on screen.

Drawing some basic primitives

- `glutInit(&argc , argv);` You use this function to initialize your glut (it takes to pointer argument).
- `glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);` You use this function to initialize your display mode if you want to specify **multiple flags you can use** |
- `glutInitWindowPosition(0,0);` You use this function to initialize window position in pixels if the position is not specified windows displayed randomly at any point on the screen

Drawing some basic primitives

- `glutInitWindowSize(500 , 500);` You use this function to initialize window size in pixels to get full Screen Size glutInitWindowSize(GetSystemMetrics(SM_CXSCREEN)
- , GetSystemMetrics(SM_CYSCREEN));
- `glutCreateWindow("Some basic primitives");` You use this function to create window with title Some basic primitives

Call back function for draw Structure

- `glutDisplayFunc(myDisplay);` you use this call back function to draw on window it takes one argument which is function pointer (which returns void and takes no arguments)

Call back function for draw Structure

- Clear your Buffers
- Resets your metrics
- Draw your objects
- Display your objects on screen

Call back function for draw Structure

- `glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);` before you drawing any thing you must clear your area (**Frame Buffer** is an area of memory which corresponds to a frame on the screen) **Frame Buffer** is drawn and then displayed so you need to set Frame Buffer to it's default values by default the buffer is cleared when the application first start up
- `GL_COLOR_BUFFER_BIT` this is a flage to color buffer that stores the color for the pixels

Call back function for draw Structure

- `glLoadIdentity();` use this function to resets all the transformations of the current metrics that you are currently in it (reset your coordinate system)
- **Afetr that draw your objects**
- `glFlush();` After everything on Buffer has been drawn so you need to call glFlush() to display this buffer on screen.

Draw objects using Open-GI

- Before you start draw need to initialize two things:-

  1- view port.

  2- projection.

Draw objects using Open-GI

- After the window is created, whenever the window is resized, maximized or minimized means the window is reshape.
- `glutReshapeFunc(reShape);` you use this call back function to reset the window size if the user reshape widow size because it might result in the distortion of your graphics on the screen it takes one argument which is function pointer (which returns void and takes two integer arguments (width and height of the window) this function is called by API [Open-GL] and those two arguments are also passed in by the API)

View Port

- View port is inside the window .
- View port is the actual rectangular (clipping area) in which the drawing are displayed so every this you draw the API will be displayed inside this area only.

Specify View Port

- `glViewport((GLint)0,(GLint)0,(GLsizei)w, (GLsizei)h);` you use this function to specify View port it takes Four arguments (the first two arguments are the x and y coordinates (relative to the bottom left corner) and the rest two parameter are width and height [you need to specify the width and height of the view port)
- So if you want your view port to take up the whole screen (that means your graphics should be displayed inside the whole screen), you need to specify x and y **(0,0)** and then the width and height should be the width and height of the window (the value of width and height the API are sent to the reShape function)

Matrices in Open_GL

- There are many different matrices in open-GL which you need to manipulate in order to work with the API (Open-GL).
- The default matrix which in the model view matrix that is used for like (rotation, scaling , translation and etc…..)
- To manipulate any matrix (Projection matrix) you need to switch to that matrix then call a function that will manipulate that matrix (projection matrix)
- `glMatrixMode(GL_PROJECTION);` you pass the parameter as the matrix that you want to switch to (projection matrix)
- `glLoadIdentity();` use this function to resets the current matrix

projection

- Projection is apart of the projection matrix.
- Projection specifies how the vertices (how the primitives you specify) are mapped inside the screen.

Specify projection

- `gluOrtho2D(-10,10,-10,10);` you use this function to specify ortho 2D projection, it takes four arguments (left, right, bottom, top) most points
- `glMatrixMode(GL_MODELVIEW);` after that you should return to ModelView
- **Now we can begin to draw**

Draw objects using Open-GI

1- call glBegin() function

2- specify our vertices.

3- call glEnd() function

Draw objects using Open-GI

- In Open-GL you specify primitives drawings **using vertices**.
- `glBegin(GL_TRIANGLES);` to draw primitives drawing you need to call glBegin function it takes one argument (which specifies what type of primitive you are going to draw)
- **Then we specify our vertices using** `glVertex2f(10,10);` which take to argument x and y with float data type
- `glEnd();` finally call glEnd() function that tells Open-GL that you have finished drawing and Open-GL can use these vertices to make the primitive with out ghEnd Open-GL will not draw any thing because it wait for the end Function.

Drawing some basic primitives

- `void changeColor() {`
- `glClearColor(1.0 , 1.0 , 0.0 , 1.0);`
- `}` default value of buffer color is black color you can use this function to change the default buffer color
  color value (0.0 – 1.0) takes four atguments (red , green , blue , alph).

Drawing some basic primitives

- `glutMainLoop();` You use this function to create program loop or execution loop