Escola de Engenharia
Universidade do Minho

Programação – Aula Teórica 12

Programação Gráfica em C

(based on Karla Fant 2014, Luis Paulo Reis 2013)

Luís Paulo Reis

lpreis@dsi.uminho.pt

Professor Associado do Departamento de Sistemas de Informação, Escola de Engenharia, Universidade do Minho, Portugal

Escola de Engenharia
Universidade do Minho

Graphical Programming in C

Outline

Objectives

- **In this lesson, you will learn:**
  - To be able to create graphical applications in C
  - To become familiar with typical graphical primitives and functions
  - To become familiar with typical graphical applications developed in C

12.1 Installation

Copy the library files at (graphics_lib.rar) on:

- **https://dl.dropboxusercontent.com/u/1910031/Aula12.rar**

to the following dev-cpp subdirectories on your PC:

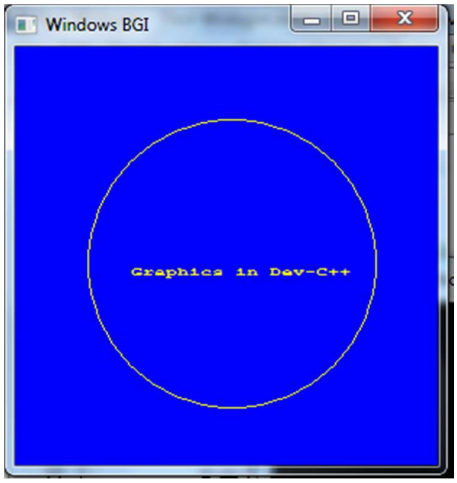
- libbgi.a file to Dev-Cpp\...\lib directory . Probably:
  - C:\Program Files (x86)\Dev-Cpp\MinGW64\x86_64-w64-mingw32\lib
- graphics.h e winbgim.h files to Dev-Cpp\...\include directory. Probably:
  - C:\Program Files (x86)\Dev-Cpp\MinGW64\x86_64-w64-mingw32\lib
- 6-ConsoleAppGraphics.template to Dev-Cpp\template

Documentation:

- http://codecutter.org/tools/winbgim/
- http://www.cs.colorado.edu/~main/bgi/dev-c++/
- http://www.cs.colorado.edu/~main/bgi/install.html
- http://www.cs.colorado.edu/~main/cs1300/doc/bgi/index.html
- https://www.youtube.com/watch?v=gibqiFtBARY
- https://www.youtube.com/watch?v=3-VWAEG8eDU

12.2 Introduction

- **Introduction Graphical Programming**
  - First Program
  - Draw a circle and text




```
/* Test Graphics with winbgim.h */

#include <winbgim.h>
#include <graphics.h>

main(int argc, char *argv[])
{
        initwindow(300, 300);              // init window graphics
        setbkcolor(1);                     // set background
        cleardevice();                     // clear screen
        setcolor(14);                      // set text color
        outtextxy(80,150,"Graphics in Dev-C++");  // print text in graphics
        circle(150,150,100);               // draw a circle
        while(!kbhit()) delay(1);          // pause screen
}
```


Conio.h

Another useful library with console versions of common I/O functions
They bypass the stdin and stdout buffers and access the console directly.

Function	Description
_kbhit	Determines if a keyboard key was pressed.
_ungetch	Puts a character back into the keyboard buffer.
_getch	Reads a character directly from the console, without echo.
_getche	Reads a character directly from the console, with echo.
_putch	Writes a character directly to the console.
_cgets	Gets a string directly from the console.
_cprintf	Formats and prints a string directly to the console.
_cputs	Outputs a string directly to the console.
_cscanf	Reads and formats values directly from the console.

Adding in Graphics Capabilities

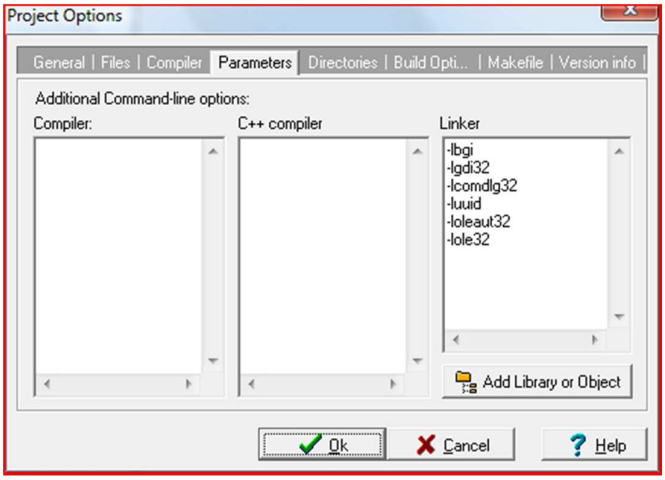
- **The graphics functions used are not automatically part of Dev-C++**
- **Download them at:**
  - graphics.h (download to C:\Dev-Cpp\include) and
  - libbgi.a (download to C:\Dev-Cpp\lib)

- **Then, tell Dev-C++ where to find the graphics library!**
- **Go to the Project menu and select Project Options**

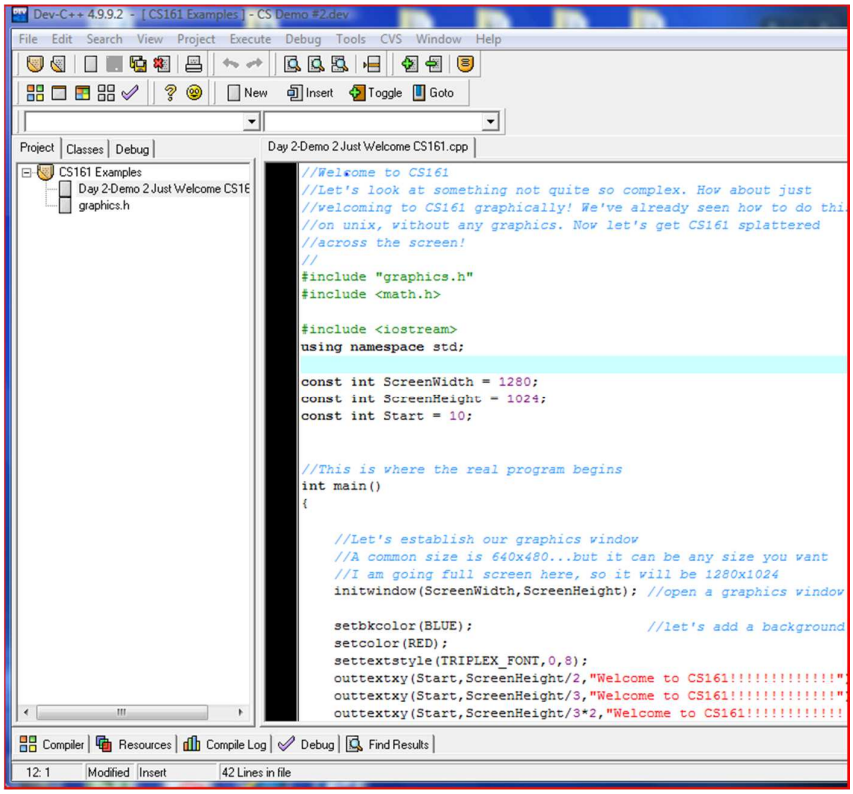
12.3 Dev C++ Projects

- **After you go to the Project menu and select Project Options**
- **Go to the Parameters tab**
- **In the Linker field, enter the following text**

  -lbgi
  -lgdi32
  -lcomdlg32
  -luuid
  -loleaut32
  -lole32
- **Click OK**



Dev C++ Apperance

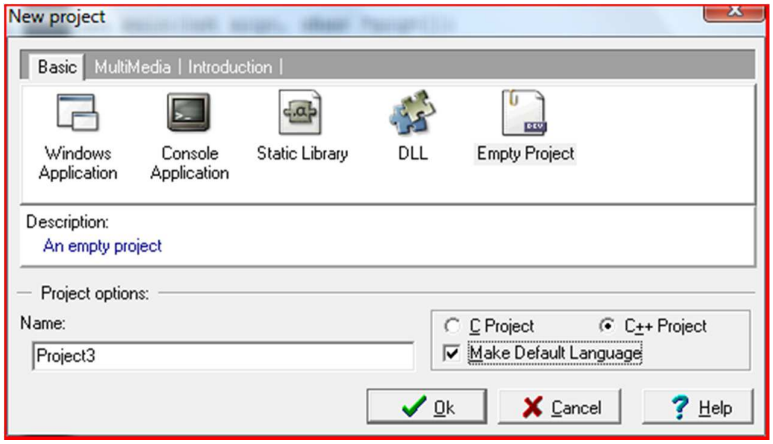


Create a new Project

- **A project is a container used to store all of the pieces needed to compile a program**
- **Go to the File menu and select New and then Project**
- **Choose Empty Project unless you use one of the projects I have posted on my website, make sure C++ project is selected**
- **This is also where you will give your project a name.**
- **If you reuse a project, you essentially will write over whatever was there, so you will probably want to start with a fresh project name. The name of your project will also be the name of your executable.**
- **Once you have entered a name of your project, click OK**
- **Dev-C++ will then ask you where you want to save your file**

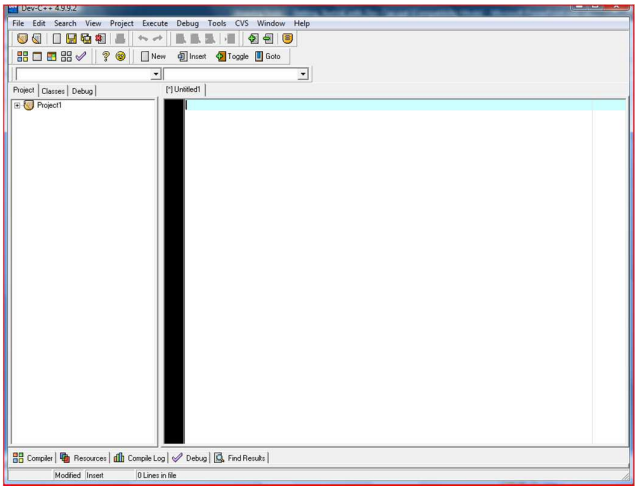
Make it a C++ Empty Project

- Select **Empty Project**
- Select **C++ Project**
- I checked "Make Default Language
- Click **OK**



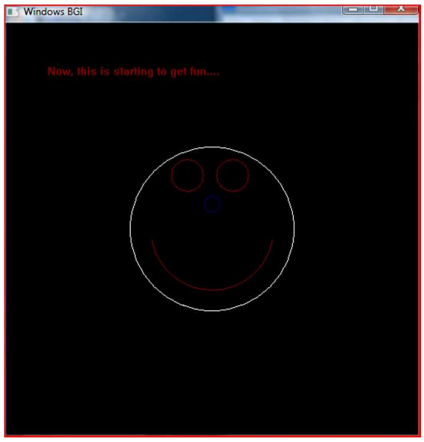
Add Source Files to Projects

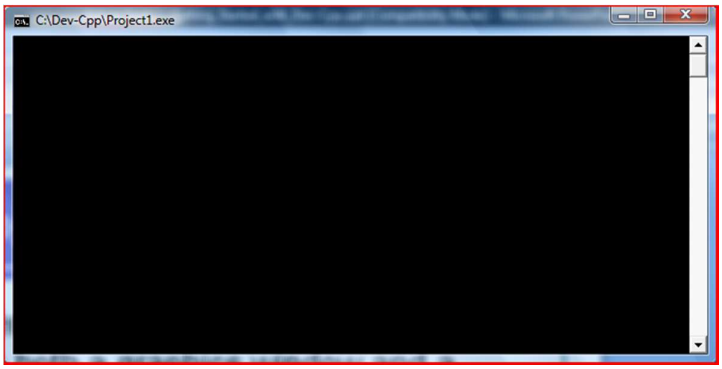
- **Some ways to add source files**
  - Go to the File menu and select New Source File
  - Or, go to the Project menu and select New File
  - I like to go to the **green +** symbol which allows me to add files to this project



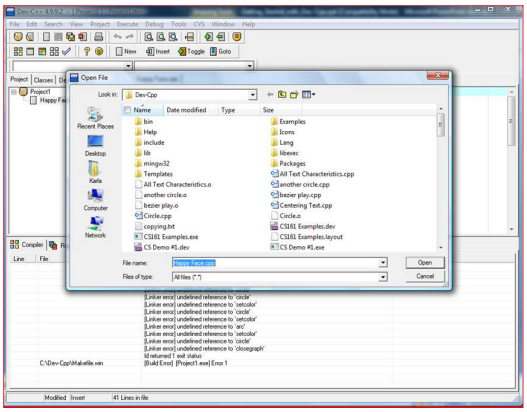
12.4 Running a Graphical Program

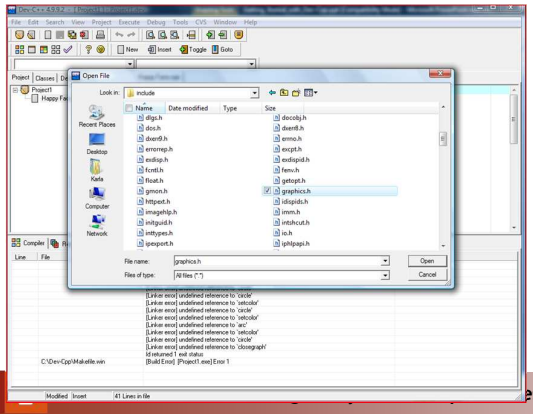
- **If there were no errors, you will get in this case both a graphics window *and a* console window:**





Placing graphics.h in the package

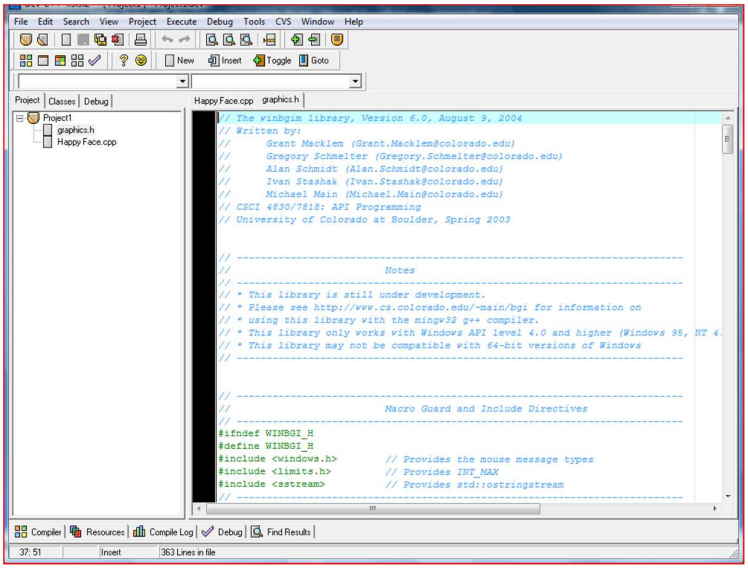




- **It is a good idea to place the graphics.h library in the project, so that you can see a summary of the functions available**
  - Go back to the **green +** symbol and select graphics.h from the include directory
  - From the include directory, select graphics.h

Environment after Selecting open

- **Notice the graphics.h in the left-hand window under the name of the project:**



By Clicking on the graphics.h file name in the left-hand window, that file is displayed now in the large window.

Don't modify this file!

12.5 Console and Graphics Windows

Console Window

- The console window is where all input and output occurs that is normal in C++
- If printf something it will go to the console window
- Most input from the keyboard works with the console window

Graphics Window:

- The graphics window is where all drawing will occur
- Printf do not work in the graphics window
- We can easily do some basic input:
  - Single character input from the keyboard
  - Mouse input

12.6 Main Graphics Features

- **Initialize the graphics window**
  initwindow(width,height);
  (the maximum width is usually 1024x768 or 1280x1024)
- **Clear the graphics window:**
  cleardevice();
- **Delay the program, so that users can see what is happening… *(in mili seconds):***
  delay(milliseconds);
- **Wait for a keyboard hit:**
  getch(); or kbhit();

Setting Graphics Attributes

- **Set Drawing Color (for lines and edges)**
  (colors typically range from 0-15; 0 is usually black and 15 is white)
  setcolor(color);
- **Set Background Color *(usually for text)***
  setbkcolor(color);
- **Set Fill Style and Color (for interiors)**
  (Pattern 0-12, 0 = empty, 1 = solid)
  setfillstyle(pattern, color)
- **Set Line Style and Thickness**
  (Style: 0 = solid, 1 = dotted, 3 = dashed)
  (Thickness is the width in terms of pixels)
  setlinestyle(style, pattern, thickness)

The Current Position for graphics

- **Origin on graphics system is Upper Left (0,0)**
  - Positive y values move DOWN
  - (x, y) coordinate data are always whole numbers

- **Setting the Current Position**
  - Move to a current position *(x,y are whole numbers)*

    moveto(x,y);
  - Move relative to the current position

    moverel(x,y);

12.7 Drawing Lines

- **Drawing a line**
  - Drawing from Current Position

    (from current position to the specified coordinate)

    lineto(x, y);
  - Drawing relative

    (a delta amount from the current position - whole number)

    linerel(deltax, deltay);
  - Drawing absolute

    (from one coordinate to another)

    linerel(from_x, from_y, to_x, to_y);

12.8 Drawing Areas

- **Drawing an unfilled Circle**

  (Given center and radius as whole numbers)

  circle (center_x, center_y, radius);

- **Drawing a Filled Circle/Ellipse**

  (Given center and radius as whole numbers)

  fillellipse(center_x, center_y, radius_x, radius_y);

- **Drawing an unfilled Rectangle**

  (given upper left and lower right corners)

  rectangle(upleft_x, upleft_y, lowrig_x, lowrig_y);

- **Drawing a Filled Rectangle**

  (given upper left and lower right corners)

  bar(upleft_x, upleft_y, lowright_x, lowright_y);

12.9 Defining Text Formatting

- **Text Formatting**
  - Set the justification

    (Horizontal: 0 = left, 1 = center, 2= right)

    (Vertical: 0 = bottom, 1 = center, 2 = top)

    settextjustify(horizontal, vertical);

  - Set the text style
    - Font: *(0-11)*
    - Direction: *0 = left to right direction*
    - Character Size: *0 = normal, 6 is very big!*

    settextstyle(font,direction, character size);

Messages in the Graphics WIndow

- **Text Output**
  - Set Text color (index ranges from 0-15):

    setcolor(index);
  - Output a message on the graphics window at the current position:

    outtext(“message to output on graphics window”);
  - Output a message on the graphics window at the given (x, y) coordinate:

    outtextxy(x, y, ”message”);

12.10 Mouse Input

Mouse Input is simples to treat with the following functions:

- **Was there a mouse click?**

  answer = ismouseclick(type);
  - Right Click: type is 513
  - Left Click: type is 516
  - Middle Click: type is 519 (mouse wheel/central button)
- **Clear the mouse click**

  clearmouseclick(type);

  (if you don’t do this you can’t get the next mouse click!)
- **What was the coordinate when the mouse click happens:**

  x = mousex(); y = mousey();

12.11 Complete Reference

- void arc (int x, int y, int stangle, int endangle, int radius);
- void bar (int left, int top, int right, int bottom);
- void bar3d (int left, int top, int right, int bottom, int depth, int topflag);
- void circle (int x, int y, int radius);
- void cleardevice (void);
- void clearmouseclick(int kind);
- void clearviewport (void);
- void closegraph (int window=ALL_WINDOWS);
- int converttorgb (int color);
- void delay (int millisec);

12.11 Complete Reference

- void detectgraph (int *graphdriver, int *graphmode);
- void drawpoly (int numpoints, int *polypoints);
- void ellipse (int x, int y, int stangle, int endangle, int xradius, int yradius);
- void fillellipse (int x, int y, int xradius, int yradius);
- void fillpoly (int numpoints, int *polypoints);
- void floodfill (int x, int y, int border);
- int getactivepage (void);
- void getarccoords (struct arccoordstype *arccoords);
- void getaspectratio (int *xasp, int *yasp);
- int getbkcolor (void);

12.11 Complete Reference

- int getch (void);
- int getcolor (void);
- int getcurrentwindow (void);
- int getdisplaycolor (int color);
- char* getdrivername (void);
- void getfillpattern (char *pattern);
- void getfillsettings (struct fillsettingstype *fillinfo);
- int getgraphmode (void);
- void getimage (int left, int top, int right, int bottom, void *bitmap);
- void getlinesettings (struct linesettingstype *lineinfo);

12.11 Complete Reference

- int getmaxcolor (void);
- int getmaxmode (void);
- int getmaxheight (void);
- int getmaxwidth (void);
- int getmaxx (void);
- int getmaxy (void);
- char* getmodename (int mode_number);
- void getmoderange (int graphdriver, int *lomode, int *himode);
- void getmouseclick(int kind, int& x, int& y);
- void getpalette (struct palettetype *palette);

12.11 Complete Reference

- int getpalettesize (void);
- int getpixel (int x, int y);
- void gettextsettings (struct textsettingstype *texttypeinfo);
- void getviewsettings (struct viewporttype *viewport);
- int getvisualpage (void);
- int getwindowheight (void);
- int getwindowwidth (void);
- int getx (void); int gety (void);
- void graphdefaults (void);
- char* grapherrormsg (int errorcode);
- int graphresult(void);

12.11 Complete Reference

- int getpalettesize (void);
- int getpixel (int x, int y);
- void gettextsettings (struct textsettingstype *texttypeinfo);
- void getviewsettings (struct viewporttype *viewport);
- int unsigned imagesize (int left, int top, int right, int bottom);
- void initgraph (int *graphdriver, int *graphmode, char *pathtodriver);
- int initwindow (int width, int height, const char* title="Windows BGI", int left=0, int top=0, bool dbflag=false, bool closeflag=true);
- int installuserdriver (char *name, int huge (*detect)(void));
- int installuserfont (char *name);

12.11 Complete Reference

- bool ismouseclick(int kind);
- int kbhit (void);
- void line (int x1, int y1, int x2, int y2);
- void linerel (int dx, int dy);
- void lineto (int x, int y);
- int mousex (void); int mousey (void);
- void moverel (int dx, int dy);
- void moveto (int x, int y);
- void outtext (char *textstring);
- void outtextxy (int x, int y, char *textstring);
- void pieslice (int x, int y, int stangle, int endangle, int radius);

12.11 Complete Reference

- void printimage (const char* title=NULL, double width_inches=7, double border_left_inches=0.75, double border_top_inches=0.75,int left=0, int right=0, int right=INT_MAX, int bottom=INT_MAX);
- void putimage (int left, int top, void *bitmap, int op);
- void putpixel (int x, int y, int color);
- void readimagefile (const char* filename=NULL, int left=0, int top=0, int right=INT_MAX, int bottom=INT_MAX);
- void rectangle (int left, int top, int right, int bottom);
- int registerbgidriver (void (*driver)(void));
- int registerbgifont (void (*font)(void));
- void registermousehandler (int kind, void h(int, int));
- void restorecrtmode (void);

12.11 Complete Reference

- void sector (int x, int y, int stangle, int endangle, int xradius, int yradius);
- void setactivepage (int page);
- void setallpalette (struct palettetype *palette);
- void setaspectratio (int xasp, int yasp);
- void setbkcolor (int color);
- void setcolor (int color);
- void setcurrentwindow (int window);
- void setmousequeuestatus(int kind, bool status=true);
- void setfillpattern (char *upattern, int color);
- void setfillstyle (int pattern, int color);

12.11 Complete Reference

- unsigned setgraphbufsize (unsigned bufsize);
- void setgraphmode (int mode);
- void setlinestyle (int linestyle, unsigned upattern, int thickness);
- void setpalette (int colornum, int color);
- void setrgbpalette (int colornum, int red, int green, int blue);
- void settextjustify (int horiz, int vert);
- void settextstyle (int font, int direction, int charsize);
- void setusercharsize (int multx, int divx, int multy, int divy);
- void setviewport (int left, int top, int right, int bottom, int clip);
- void setvisualpage (int page);

12.11 Complete Reference

- void setwritemode (int mode);
- int showerrorbox (const char *message);
- int swapbuffers (void);
- int textheight (char *textstring);
- int textwidth (char *textstring);
- void writeimagefile (const char* filename=NULL,double width_inches=7, double border_left_inches=0.75, double border_top_inches=0.75, int left=0, int top=0, int right=INT_MAX, int bottom=INT_MAX);

Questões?

Programação – Aula Teórica 12

Programação Gráfica em C

Luís Paulo Reis
lpreis@dsi.uminho.pt
Professor Associado do Departamento de Sistemas de Informação, Escola de Engenharia, Universidade do Minho, Portugal