CS250 Intro to CS II

Spring 2014

Dark GDK
Sprites

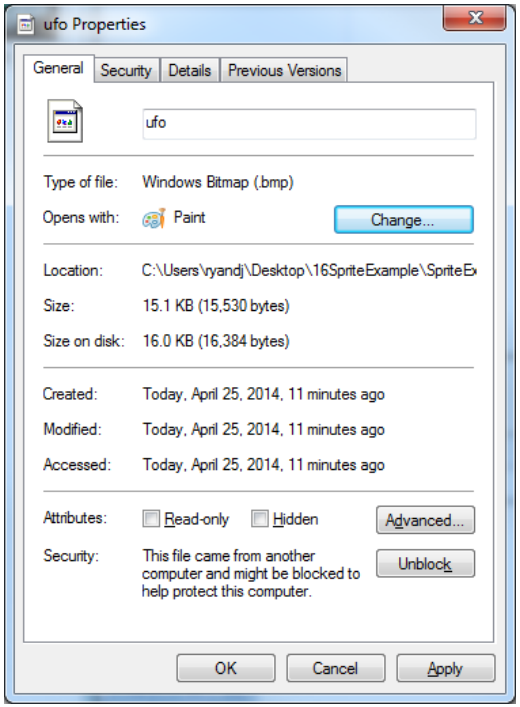
- Up to now, you've written console apps and Dark GDK apps that draw primitive shapes
- It's time to use images that have been created with a software app (e.g. Paint, Photoshop) or captured with a digital device (e.g. scanner, camera)

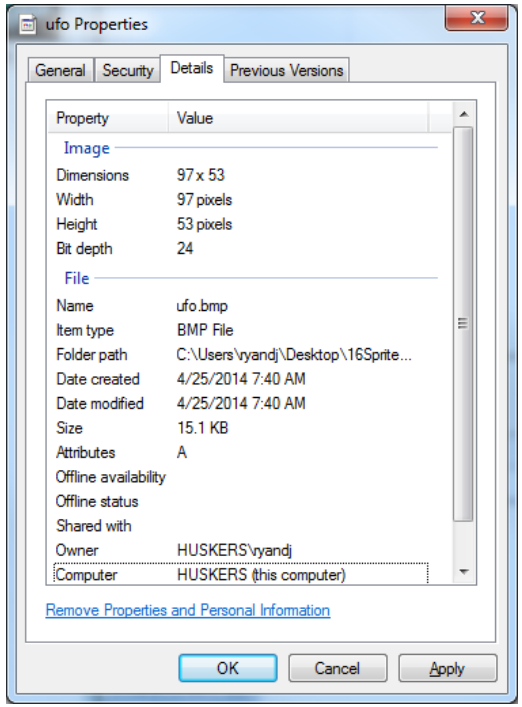
ufo.bmp

Images

- Images are commonly saved as bitmaps
- Dark GDK provides functions for loading, displaying, and modifying bitmaps
- bitmap – data that describes every pixel in an image
- Dark GDK has a function dbLoadBitmap that loads a bitmap file into memory
- Acceptable file formats are: .bmp, .jpg, .tga, .dds, .dib, or .png

Images


ufo Properties
General
Security
Details
Previous Versions
ufo
Type of file: Windows Bitmap (.bmp)
Opens with: Paint
Change...
Location: C:\Users\ryandj\Desktop\16SpriteExample\SpriteEx
Size: 15.1 KB (15,530 bytes)
Size on disk: 16.0 KB (16,384 bytes)
Created: Today, April 25, 2014, 11 minutes ago
Modified: Today, April 25, 2014, 11 minutes ago
Accessed: Today, April 25, 2014, 11 minutes ago
Attributes: Read-only Hidden
Advanced...
Security: This file came from another computer and might be blocked to help protect this computer.
Unblock
OK
Cancel
Apply

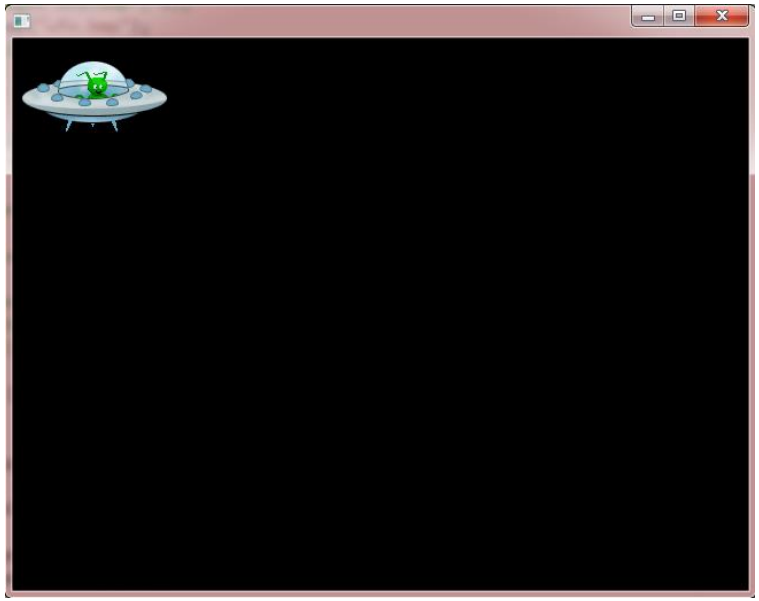

ufo Properties
General
Security
Details
Previous Versions
Property
Value
Image
Dimensions 97 x 53
Width 97 pixels
Height 53 pixels
Bit depth 24
File
Name ufo.bmp
Item type BMP File
Folder path C:\Users\ryandj\Desktop\16Sprite...
Date created 4/25/2014 7:40 AM
Date modified 4/25/2014 7:40 AM
Size 15.1 KB
Attributes A
Offline availability
Offline status
Shared with
Owner HUSKERS\ryandj
Computer HUSKERS (this computer)
Remove Properties and Personal Information
OK
Cancel
Apply

Sample Dark GDK Bitmap Program

```
#include "DarkGDK.h"

void DarkGDK ()
{
  // Load and display image
  dbLoadBitmap ("ufo.bmp");

  dbWaitKey ();
}
```



Where to place images?

- Consider a Studio solution called CS250 with a project called Sprites.
- Bitmaps are placed in the Sprites folder unless otherwise specified.
- In other words, the image files are placed in the same folder as your driver.cpp file.

Dark GDK
Sprites

- A sprite is a graphic image used in serious game development.
- In DarkGDK you need to:
  1. Load the images into memory using dbLoadImage
  2. Display the images to the screen using dbSprite

dbLoadImage

- The format for dbLoadImage is:

  void dbLoadImage (char* szFilename, int iImage)

- szFilename is the name of the file
- iImage is a number that you assign to the image. You will use this number with dbSprite to display the image to the screen

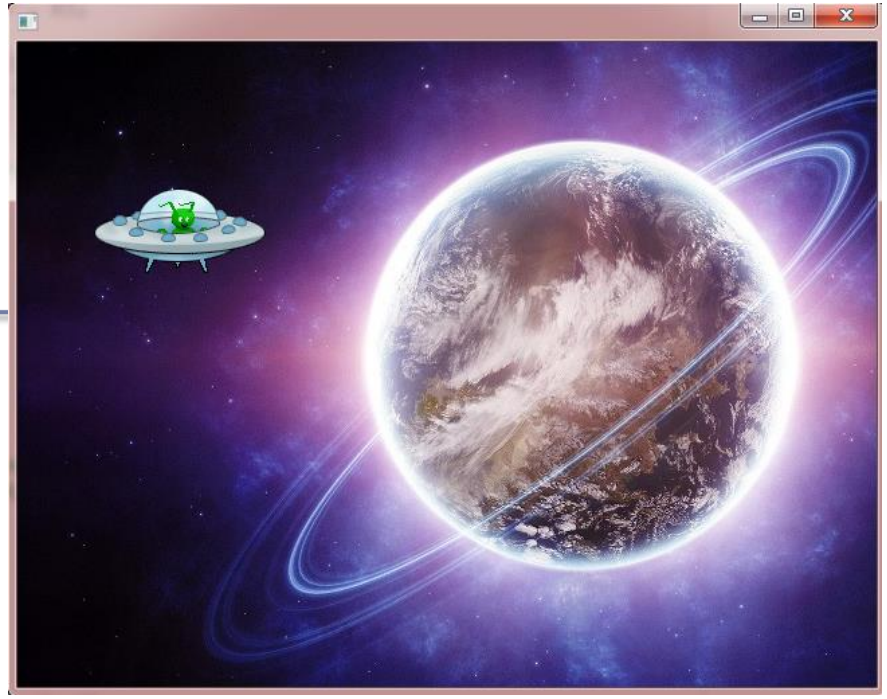
dbSprite

- The format for dbSprite is:

  void dbSprite (int iSprite, int iX, int iY, int iImage)

- iSprite is the number that you are assigning to the sprite
- x, y are the screen coordinates for the upper left corner of the image
- iImage is the number of the image that you want to use for the sprite

Simple Sprite Program

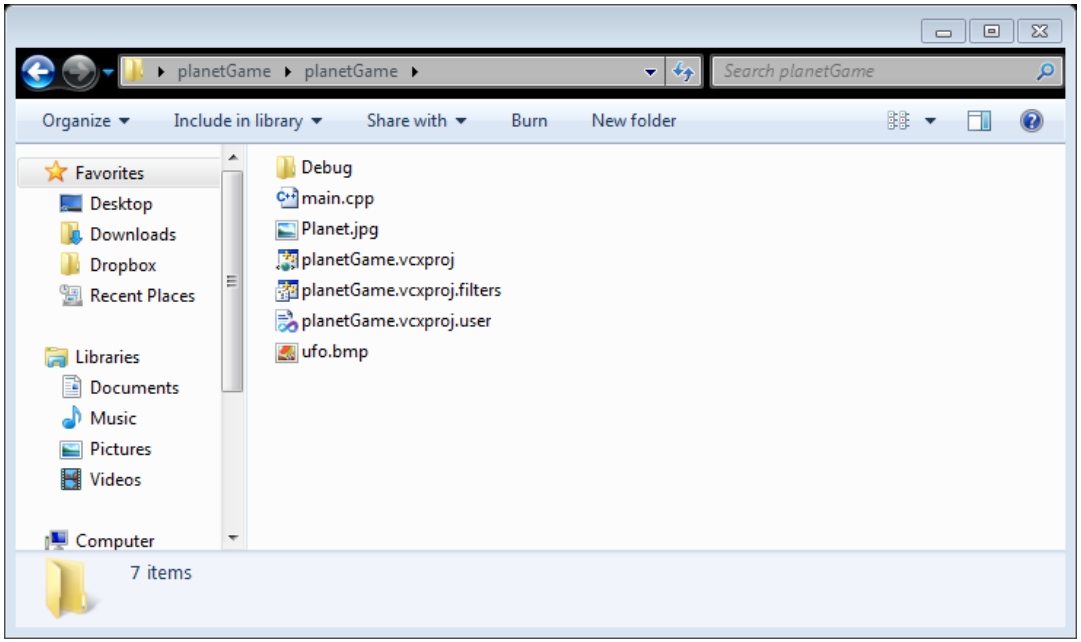


```
#include "DarkGDK.h"
#include "Sprite.h"

void DarkGDK (void)
{
  dbLoadImage("Planet.jpg", 1);
  dbLoadImage("ufo.bmp", 2);
  dbSprite(1, 0, 0, 1);
  dbSprite(2, 50, 90, 2);
  dbWaitKey ();
}
```

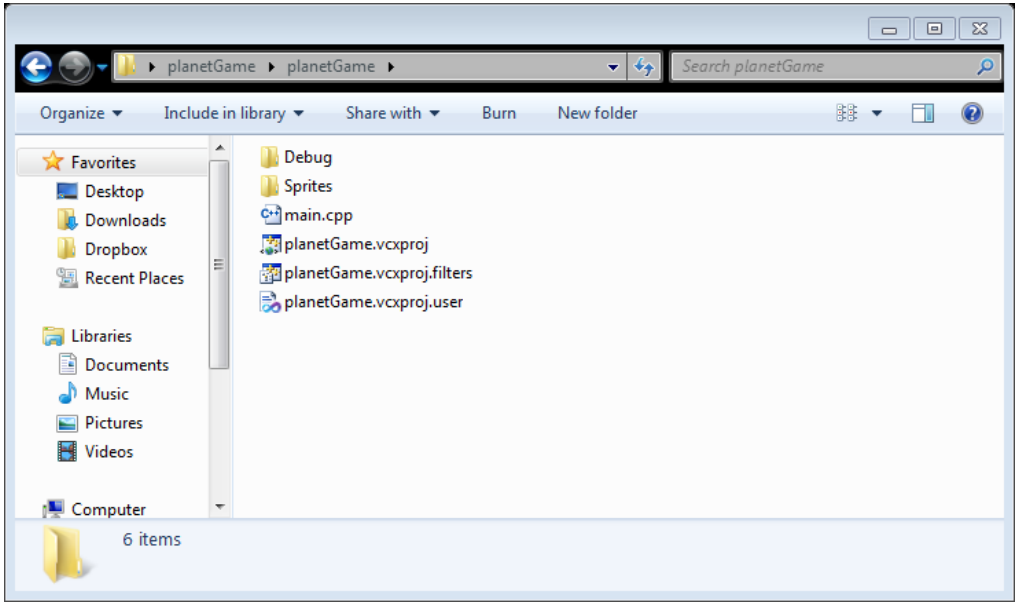
Location of Images

- Images are in the same location as .cpp
- What would be a better organization?



Location of Images

- Create a folder (Sprites)
- What would need to change in the code?



Location of Images

```
#include "DarkGDK.h"
#include "Sprite.h"

void DarkGDK (void)
{
  dbLoadImage("Sprites/Planet.jpg", 1);
  dbLoadImage("Sprites/ufo.bmp", 2);
  dbSprite(1, 0, 0, 1);
  dbSprite(2, 50, 90, 2);
  dbWaitKey ();
}
```

Moving Sprites

- Sprites can be moved by changing the x and y coordinates inside of dbSprite
- One way you might want to do this is by having the user control a sprite using the arrow keys
- You can capture the key input using:
  - dbUpKey(), dbDownKey(), dbLeftKey(), and dbRightKey()

Moving UFO Sprite Example

- Run the UFO Sprite example found in the public folder on turing
- Let's look at the code

Useful Sprite Functions

- DarkGDK sprite functions
- Let's go to the documentation and look at the Sprite functions

Dark GDK
Sprites

- Open up SpriteExample, which you will find in the public folder on Turing
- Uncomment the documented code and comment out dbLoadBitmap (“Sprites/ufo.bmp”);
- Look at the code and try to figure out what it will do before trying to run the code

Problem #1

- Modify the SpriteExample as follows:
  1. Create an array of pointers to Sprites
  2. Dynamically allocate space for 25 balls. Make sure the balls are placed somewhere on the screen. The dimensions of the Sprites are 50x50.
  3. Display the Sprites on the screen.
  4. When the program is terminated, free all dynamically allocated Sprites.
  5. Using the debugger, check to see that all space is freed.

Problem #2

- Load up the deck of cards in the Cards folder
- Display one card every second