CS 102 Lab 5: Release the Gifs

Specification

This program will allow the user to type in a search word and then search the Giphy website to find a gif related to the word they searched. Once it finds the gif it will animate it in a seperate window, which will scale with the gif's size.

The API for giphy can be found for free on the mashape marketplace

Analysis

We will use the `market.mashape.com/explore`mashape market to get the API for the Giphy website from `market.mashape.com/giphy/giphy`. We will also use a json API from `https://github.com/nlohmann/json` to parse the json data and also the CURL API, from `curl.haxx.se/libcurl` to get the gif file. The gif will be animated using FLTK and Cairo from `www.fltk.org/doc-1.3/group__group__cairo.html`

Each url is a hyperlink which leads to a webpage where you can learn more about each part of the program

Design

- labh
  - Contains all shared headers and variables
- makeSearchWindow
  - Makes a window with a search box that people can use to find Gif
- getGifInfo
  - Get data, in json format, from Giphy API from mashape specific to the users keywords
- extractGifInfo
  - Given a string in json format with all the info about the gif and return the information
- loadImage
  - Turns the gif file into sequenced photographs
- cbDisplay
  - Uses given keyword, finds appropriate gif, and displays the gif
- cbanimate
  - animates the sequenced photos
- main
  - Use console to test getting and displaying a gif then using a GUI to let user choose their own file

Each of the implementation slides, and therefore each .cpp file, has one function in it. The function shares the name of the file aswell.

Implementation lab.h

```
#include <config.h>
#include <cstdlib>
#include <iostream>
#include <fstream>
#include <FL/Fl_Cairo_Window.H>
#include <FL/Fl_Input.H>
#include <FL/Fl_Button.H>
#include <FL/Fl_Box.H>
#include <FL/Fl_GIF_Image.H>
#include <fstream>
#include <sstream>
#include <iomanip>
struct GifInfo{std::string url;int width;int height;int frames;};
extern GifInfo gi;const int width = 300;const int height = 300;
Fl_Cairo_Window * makeSearchWindow();
Fl_Cairo_Window * makeDisplayWindow(int,int);
void loadImages(std::string g,int frames);
void cbDisplay(Fl_Button*,void*);
extern Fl_Cairo_Window * dw;extern Fl_Box * g;
extern Fl_Input * r;extern Fl_Cairo_Window * cg;
GifInfo extractGifInfo(std::string g);
std::string getGifInfo(std::string s);
extern Fl_GIF_Image ** images;void cbAnimate(void*);
```

- Contains all variables and headers files used by other functions

Implementation makeSearchWindow.cpp

```
#include "lab.h"
Fl_Cairo_Window * cw;
Fl_Input * r;
Fl_Button * b;

Fl_Cairo_Window * makeSearchWindow(){
    cw = new Fl_Cairo_Window(width,height);
    cw->label("Search For A Gif");
    cw->color(fl_rgb_color(74,189,172));
    r = new Fl_Input(.5*width,.25*height,.25*width,.1*height);
    r->label("Keyword: ");
    b = new Fl_Button(.5*width,.5*height,.25*width,.1*height);
    b->label("Display");
    b->color(fl_rgb_color(247,183,51));
    b->callback((Fl_Callback*) cbDisplay);
    return cw;}
```

- Make a Window to obtain keyword
  - Makes a Window called "Search For A Gif"
  - Makes type box for keyword
  - Makes button to send keyword to other functions

Implementation getGifInfo.cpp

```
#include "lab.h"
#include <curl/curl.h>
const std::string url = "https://giphy.p.mashape.com";
const std::string key =
"X-Mashape-Key: N2gdkwRwq8mshigJOyEc4T1ibhjqp16Oc4SjsnPVK3HCrfVhMo";
const std::string js = "Accept: application/json";
size_t handleData(void* c, size_t s, size_t n, void* g)
{*static_cast<std::string*>(g) += static_cast<char*>(c);return s * n;}
std::string getGifInfo(std::string s)
{std::string j;    struct curl_slist *slist1 = NULL;
    slist1 = curl_slist_append(slist1,key.c_str());
    slist1 = curl_slist_append(slist1,js.c_str());
    std::string q = url +
    "/v1/gifs/search?api_key=dc6zaTOxFJmzC&limit=1&q="
     + s;
    CURL* hnd = curl_easy_init();
    curl_easy_setopt(hnd,CURLOPT_URL,q.c_str());
    curl_easy_setopt(hnd,CURLOPT_HTTPHEADER,slist1);
    curl_easy_setopt(hnd,CURLOPT_WRITEFUNCTION,handleData);
    curl_easy_setopt(hnd,CURLOPT_WRITEDATA,&j);
    curl_easy_perform(hnd);
    while(j.back() != '}') j.pop_back();return j;
}
```

- goal: get the json infro from Giphy API
  - expect string parameter with keyword
  - use the CURL library to retrieve the data
  - return the json data

Implementation extractGifInfo.cpp

```
#include "lab.h"
#include <iostream>
#include "json.hpp"
#include <sstream>
GifInfo gi;using nlohmann::json; GifInfo extractGifInfo(std::string g)
{std::string s; auto j = json::parse(g); for(auto &e : j["data"])
    {   s += e["images"]["original"]["url"]; s += " ";
        s += e["images"]["original"]["width"]; s += " ";
        s += e["images"]["original"]["height"]; s += " ";
        s += e["images"]["original"]["frames"];}
    std:: istringstream iss(s);
    iss >> gi.url >> gi.width >> gi.height >> gi.frames;return gi;
}
```

- goal: extract from the json info about the gif
  - accepts string parameter with data about gif in json format
  - use parse function from json api from github
  - returns: width, height, frames, and url


```
– "original": {
        "url": "http://media4.giphy.com/media/ivelTBasMg67S/giphy.gif",
        "width": "500",
        "height": "350",
        "size": "1004245",
        "frames": "31",
        "mp4": "http://media.giphy.com/media/ivelTBasMg67S/giphy.mp4"
      },
```

Implementation makeDisplayWindowDoc.cpp

```
#include "lab.h"
Fl_Cairo_Window * dw;
Fl_Box * g;

Fl_Cairo_Window * makeDisplayWindow(int width, int height){
    dw = new Fl_Cairo_Window(width,height);
    dw->label("Display of Gif");
    dw->color(fl_rgb_color(74,189,172));
    g = new Fl_Box(FL_FLAT_BOX, 0,0,width,height,"");
    return dw;}
```

- goal:Make a window that displays the gif
  - Make a window call "Display of Gif"
  - Makes a with dimensions of the gif, for the gif to be put in

Implementation loadImage.cpp

```
#include "lab.h"
Fl_GIF_Image ** images;
void loadImages(std::string g,int frames)
{
    std::string cmd ="rm images/*";
    system(cmd.c_str());
    cmd = "convert -coalesce " + g + " images/g%03d.gif";
    system(cmd.c_str());
    images = new Fl_GIF_Image* [frames];
    for(int i = 0; i < frames; i++)
    {
        std::ostringstream oss;
        oss << std::setfill('0') << std::setw(3) << i;
        std::string s = "images/g" + oss.str() + ".gif";
        images[i] = new Fl_GIF_Image(s.c_str());
        dw -> redraw();
    }

}
```

- goal:Turn the gif file into a sequence of photos
  - first empty image folder
  - convert each frame into its own photo saved in the image folder
  - then display each image starting form 000 up to 999

Implementation cbDisplay.cpp

```
#include "lab.h"
void cbDisplay(Fl_Button*,void*)
{
    std::string s = r->value();
    std::string j = getGifInfo(s);
    GifInfo gi = extractGifInfo(j);
    //std::cout << "W:" << gi.width << std::endl;
    //std::cout << "H:" << gi.height << std::endl;
    std::string cmd = "curl " + gi.url +" > 200.gif";
    system(cmd.c_str());
    if(dw) dw->hide();
    makeDisplayWindow(gi.width,gi.height) -> show();
    loadImages("200.gif",gi.frames);
    Fl::add_timeout(1,cbAnimate,&gi.frames);

}
```

- goal:Get the Gif and display it in window
  - places the string in the GUI into the string s
  - which is put into getGifInfo
  - the result of getGifInfo is put in extractGifInfo
  - we create a string to download the specific gif, and save it
  - creates window off gif data, and won't if one window already exists

Implementation cbAnimate.cpp

```
#include "lab.h"
void cbAnimate(void*)
{
    static int x = 0;
    g->image(images[x]);
    x++;
    if(x == gi.frames) x=0;
    dw -> redraw();
    Fl::repeat_timeout(1.0/24,cbAnimate);
}
```

- goal:Animate the sequenced photos
  - displays image with value x
  - As x increases by 1 so does the image value
  - if the image value exceeds the number of frames it will reset
  - it will redraw between each frame 24 times a second
  - Thus the gif is displayed at 24 fps

Implementation main.cpp

```
#include "lab.h"


int main()
{
    makeSearchWindow() -> show();
    Fl::run();
    return 0;
}
```

- Retrieve the gif file
- Display it

Test

- The user can search whatever they want
  - If the user searches for "fish" they will get a fish related gif
    - The displayed results can be seen in output1.avi
  - if the user searches for "monty+oum" you will get another related gif
    - The displayed results can be seen in output2.avi

The user cannot type in spaces, but can use "+" as an alternative