Dr Duck

1.0.0

Generated by Doxygen 1.9.1

Chapter 1

Dr Duck

- CS162 projects.
- This is a solo project for the course CS162 - Introduction to Computer Science II.
- Visualize data structures.
- In this project, I will do a data visualization application.

1.1 FEATURES

1.1.0.1 Static array / dynamic array / singly, doubly, circular linked list / stack / queue

- Init from file
- Randomized data
- Insert / delete / search / update / push / pops
- Highlight code
- Custom

1.1.0.2 Run step by step

1.1.0.3 Speed up and slow down

1.1.0.4 Sound

1.1.0.5 Graphics

1.1.0.6 Open with file

- Data is storing in folder saving/

1.2 REQUIRED

- SDL2
- SDL_image
- SDL_ttf
- https://github.com/nlohmann/json
- CMAKE 3.12+
- makefile
- C++17

1.3 INSTALL

- At first you need to install C++17, makefile, CMake:
```
sudo apt update
sudo apt install build-essential
sudo apt install make
sudo apt install cmake
```
- Install SDL2 and external libraries
```
sudo apt install libsdl2-dev
sudo apt install libsdl2-ttf-dev
sudo apt install libsdl2-image-dev
git clone https://github.com/nlohmann/json.git
cd json
mkdir build
cd build
cmake ..
make
sudo make install
```
- After installed, download the repository and call make.
```
git clone https://github.com/qvanle/drDuck
cd drDuck
make
```

1.4 Usage

1.4.1 1. After call make, a window will pop up

- If you don't like the sound you can turn off it (the button is on the top-left of screen).
- Click "start" button to continue.
- Tutorial and donate are still in developing.

1.4.2 2. You have 3 type of data structures

1. Array.
2. Linked list.
3. In and out order (stack and queue).

1.4.2.1 Array

1. Static array.
2. Dynamic array.

1.4.2.2 Linked list

1. Singly linked list.
2. Doubly linked list.
3. Circular linked list.

1.4.2.3 I/O order

1. Stack.
2. Queue.

1.4.3 3. Open file

- After press button "open file" there will a box pop up require to enter file name.
- This file must be store in folder saving, if there don't exist that file, the window unable to do anything.
- To create new file, you have to save it with file extension "json"
- Samples are in saving/

1.4.4 4. Working screen

1.4.4.1 After you have choose data structures or open file, this screen will be displayed.

1.4.4.2 Move the cursor to the right, operator bar will be appeared.

1.4.4.3 Move the cursor to the bottom, play bar will be appeared.

1. First button will slow down animation
2. Second button (still in developing)
3. Third button will Pause/Continue animation
4. Fourth button will go to next step (when the animation is pause)
5. Last button will speed up the animation

1.4.5 5. Opeartor bar

1.4.5.1 New operator

- Maximum size of static and dynamic array is 12, otherwise it's 10.
- If the input greater than maximum size, it will ignore the remainder.

1.4.5.2 Insert operator

- If the size is maximum, it will do nothing.
- In stack and queue mode, this will be call push.

1.4.5.3 Delete operator

- If the size is equal 0, it will do nothing.
- In stack and queue mode, this will be call pop.

1.4.5.4 Update operator

1.4.5.5 Search operator

1.5 Demonstration

1.6 Documentation

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 GLOBAL Namespace Reference

store all global variables

Variables

- const char * GraphicsFolder = "asset/graphics/"

  path to the folder that store all graphics
- const char * BackgroundFolder = "asset/graphics/"

  path to the folder that store all background asset
- const char * ButtonFolder = "asset/graphics/"

  path to the folder that store all button asset
- const char * AttributeFolder = "asset/attribute/"

  path to the folder that store all custom attribute of graphics, sound, etc
- const char * AtrbScreens = "asset/attribute/screens/"

  path to the folder that store all custom attribute of screens
- const char * AtrbButtons = "asset/attribute/buttons/"

  path to the folder that store all custom attribute of buttons
- const std::string AtrbDT = "asset/attribute/DataStructures/"

  path to the folder that store all custom attribute of data structures
- const char * FontsFolder = "asset/fonts"

  path to the folder that store all custom attribute of fonts
- const std::string AtrbInputBox = "asset/attribute/InputBox/"

  path to the folder that store all custom attribute of input box
- const std::string AtrbScript = "asset/attribute/script/"

  path to the folder that store all custom attribute of script
- const int WAITING = 800

  time to wait for the next action
- const std::string SoundFolder = "asset/sound/"

  path to the folder that store all custom attribute of sound

6.1.1 Detailed Description

store all global variables

6.1.2 Variable Documentation

6.1.2.1 AtrbButtons

```
const char * GLOBAL::AtrbButtons = "asset/attribute/buttons/"  [extern]
```

path to the folder that store all custom attribute of buttons

Definition at line 28 of file SYSTEM.cpp.

6.1.2.2 AtrbDT

```
const std::string GLOBAL::AtrbDT = "asset/attribute/DataStructures/"  [extern]
```

path to the folder that store all custom attribute of data structures

Definition at line 40 of file SYSTEM.cpp.

6.1.2.3 AtrbInputBox

```
const std::string GLOBAL::AtrbInputBox = "asset/attribute/InputBox/"  [extern]
```

path to the folder that store all custom attribute of input box

Definition at line 36 of file SYSTEM.cpp.

6.1.2.4 AtrbScreens

```
const char * GLOBAL::AtrbScreens = "asset/attribute/screens/"  [extern]
```

path to the folder that store all custom attribute of screens

Definition at line 24 of file SYSTEM.cpp.

6.1.2.5 AtrbScript

```
const std::string GLOBAL::AtrbScript = "asset/attribute/script/"  [extern]
```

path to the folder that store all custom attribute of script

Definition at line 44 of file SYSTEM.cpp.

6.1.2.6 AttributeFolder

```
const char * GLOBAL::AttributeFolder = "asset/attribute/"  [extern]
```

path to the folder that store all custom attribute of graphics, sound, etc

Definition at line 20 of file SYSTEM.cpp.

6.1.2.7 BackgroundFolder

```
const char * GLOBAL::BackgroundFolder = "asset/graphics/"  [extern]
```

path to the folder that store all background asset

Definition at line 11 of file SYSTEM.cpp.

6.1.2.8 ButtonFolder

```
const char * GLOBAL::ButtonFolder = "asset/graphics/"  [extern]
```

path to the folder that store all button asset

Definition at line 15 of file SYSTEM.cpp.

6.1.2.9 FontsFolder

```
const char * GLOBAL::FontsFolder = "asset/fonts"  [extern]
```

path to the folder that store all custom attribute of fonts

Definition at line 32 of file SYSTEM.cpp.

6.1.2.10 GraphicsFolder

```
const char * GLOBAL::GraphicsFolder = "asset/graphics/"  [extern]
```

path to the folder that store all graphics

Definition at line 6 of file SYSTEM.cpp.

6.1.2.11 SoundFolder

```
const std::string GLOBAL::SoundFolder = "asset/sound/"  [extern]
```

path to the folder that store all custom attribute of sound

Definition at line 54 of file SYSTEM.cpp.

6.1.2.12 WAITING

```
const int GLOBAL::WAITING = 800  [extern]
```

time to wait for the next action

for step to step feature

Definition at line 50 of file SYSTEM.cpp.

6.2 RANDOM Namespace Reference

store all randomize object

Functions

- void init ()
  initialize random number generator
- int getInt (int l, int r)
  get random integer in range [l, r]
- double getDouble (double l, double r)
  get random double in range [l, r]

Variables

- std::mt19937 rng
  random number generator

6.2.1 Detailed Description

store all randomize object

6.2.2 Function Documentation

6.2.2.1 getDouble()

```
double RANDOM::getDouble (
            double l,
            double r )
```

get random double in range [l, r]

Parameters

l	double, left bound
r	double, right bound

Definition at line 84 of file SYSTEM.cpp.

```
85 {
86     return std::uniform_real_distribution<double> (l, r)(rng);
87 }
```

6.2.2.2 getInt()

```
int RANDOM::getInt (
            int l,
            int r )
```

get random integer in range [l, r]

Parameters

l	interger, left bound
r	integer, right bound

Definition at line 75 of file SYSTEM.cpp.

```
76 {
77     return std::uniform_int_distribution<int> (l, r)(rng);
78 }
```

6.2.2.3 init()

```
void RANDOM::init ( )
```

initialize random number generator

call this before using random

Definition at line 66 of file SYSTEM.cpp.

```
67 {
68     rng = std::mt19937(std::chrono::steady_clock::now().time_since_epoch().count());
69 }
```

6.2.3 Variable Documentation

6.2.3.1 rng

```
std::mt19937 RANDOM::rng  [extern]
```

random number generator

Definition at line 60 of file SYSTEM.cpp.

Chapter 7

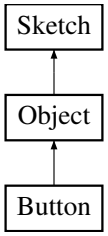
Class Documentation

7.1 Button Class Reference

class that represents a button. Button is just a object that when is triggered, it will do something.

`#include <Button.hpp>`

Inheritance diagram for Button:



Public Member Functions

- Button ()
- std::string getAction ()
- bool isChosen (int x, int y)
- void init (const char ∗name)
- void init (const char ∗dir, const char ∗name)
- void init (const json &mem)
- void setRenderer (SDL_Renderer ∗const &r)
- void render (bool update)
- void render ()
- void Delete ()
- void clearTextures ()
- char ∗const & getNextScreen ()
- void setDataStructure (std::string s)
- std::string getDataStructure ()
- ∼Button ()
- void init (const json &mem, SDL_Renderer ∗&r)

  init this object by json file
- void setCoor (int x, int y, int w, int h)

set coordinate of this object

- void setCoor (SDL_Rect key)

  set coordinate of this object
- void setX (int x)

  set x coordinate of this object
- void setY (int y)

  set y coordinate of this object
- void setW (int w)

  set width of this object
- void setH (int h)

  set height of this object
- const SDL_Rect & getCoor ()

  return coordinate of this object
- bool isLiesInside (int x, int y)

  return true if a point lies inside this object
- bool isLiesInside (int x, int y, int w, int h)

  return true if a rectangle inside this object
- bool isLiesInside (SDL_Rect rect)

  return true if a rectangle inside this object
- void addX (int k)

  add k to x coordinate
- void addY (int k)

  add k to y coordinate
- void addW (int k)

  add k to width
- void addH (int k)

  add k to height
- void show ()

  show this object
- void hide ()

  hide this object
- bool isVisible ()

  return true if this object is visible
- void setTextures (const json &mem)

  load textures from json file
- void pickTexure (int k)

  choose top texture
- int size ()

  return number of textures
- void moveTo (int x, int y, double time)

  move this object to (x, y) in time seconds
- void setRender (SDL_Renderer ∗&r)

  set render
- void addChar (char ch)

  typing text
- void popChar ()

  erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create
- void setText (std::string s)

  set text to be s
- void setTextColor (int r, int g, int b)

set text color to be (r, g, b)

- const std::string & getText ()

  return text

- void setColor (SDL_Color c)

  set background color to be c

- void setColor (int r, int g, int b)

  set background color to be (r, g, b)

- void setColor (int r, int g, int b, int a)

  set background color to be (r, g, b a)

- void setInCenterX ()

  align text texture align x coordinate of text texture to be in the center of the background texture

- void setOnLeftSideX ()

  align text texture align x coordinate of text texture to be in the left side of the background texture

- void setOnRightSideX ()

  align text texture align x coordinate of text texture to be in the right side of the background texture

- void setInCenterY ()

  align text texture align y coordinate of text texture to be in the center of the background texture

- void setOnLeftSideY ()

  align text texture align y coordinate of text texture to be in the top side of the background texture

- void setOnRightSideY ()

  align text texture align y coordinate of text texture to be in the bottom side of the background texture

- void align ()

  align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

- void setBorder (int w, int r, int g, int b, int a)

  set border

- void setBorderColor (int r, int g, int b)

  set border color

- void FillWithColor ()

  fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

- void FillWithColor (SDL_Color c)

  fill background color with color C

- void highlight ()

  hightlight the sketch this function will change color of background to invert color

- void unHighlight ()

  unhightlight the sketch this function will change color of background to normal color

- bool isLieInside (int x, int y)

  determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

Protected Member Functions

- void clearTexture (int k)

  clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault

- void initColor (const json &mem)

  init color from json

- void initFont (const json &mem)

  init font from json

- void initBorder (const json &mem)

  init border from json

- void createTextTexture ()

  create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

7.1.1 Detailed Description

class that represents a button. Button is just a object that when is triggered, it will do something.

Definition at line 18 of file Button.hpp.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 Button()

```
Button::Button ( )
```

Definition at line 3 of file Button.cpp.

```
4 {
5     ren = nullptr;
6     msg = nullptr;
7     argc = 0;
8     argv.clear();
9 }
```

7.1.2.2 ~Button()

```
Button::~Button ( )
```

Definition at line 118 of file Button.cpp.

```
119 {
120     Delete();
121 }
```

7.1.3 Member Function Documentation

7.1.3.1 addChar()

```
void Sketch::addChar (
             char ch )  [inherited]
```

typing text

Parameters

ch	character that will be add to the end of the text add a character to the end of the text after that new text texture will be created

Definition at line 101 of file Sketch.cpp.

```
102 {
103     text = text + ch;
104     createTextTexture();
105 }
```

7.1.3.2 addH()

```
void Object::addH (
              int k )  [inherited]
```

add k to height

Definition at line 308 of file Object.cpp.

```
309 {
310     coor.h += k;
311 }
```

7.1.3.3 addW()

```
void Object::addW (
              int k )  [inherited]
```

add k to width

Definition at line 301 of file Object.cpp.

```
302 {
303     coor.w += k;
304 }
```

7.1.3.4 addX()

```
void Object::addX (
              int k )  [inherited]
```

add k to x coordinate

Definition at line 287 of file Object.cpp.

```
288 {
289     coor.x += k;
290 }
```

7.1.3.5 addY()

```
void Object::addY (
              int k )  [inherited]
```

add k to y coordinate

Definition at line 294 of file Object.cpp.

```
295 {
296     coor.y += k;
297 }
```

7.1.3.6 align()

```
void Sketch::align ( )  [inherited]
```

align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

Definition at line 761 of file Sketch.cpp.

```
762 {
763
764     if(textAlignX == 1) setOnLeftSideX();
765     if(textAlignX == 2) setInCenterX();
766     if(textAlignX == 3) setOnRightSideX();
767
768     if(textAlignY == 1) setOnLeftSideY();
769     if(textAlignY == 2) setInCenterY();
770     if(textAlignY == 3) setOnRightSideY();
771 }
```

7.1.3.7 clearTexture()

```
void Sketch::clearTexture (
            int k )  [protected], [inherited]
```

clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault

Parameters

k	integer, index of textures, 0 will be background, 1 will be text if tes[k] is nullptr, do nothing call SDL_DestroyTexture and after that set tes[k] to be nullptr

Definition at line 37 of file Sketch.cpp.

```
38 {
39     if(tes[k] == nullptr) return ;
40     SDL_DestroyTexture(tes[k]);
41     tes[k] = nullptr;
42 }
```

7.1.3.8 clearTextures()

```
void Button::clearTextures ( )
```

7.1.3.9 createTextTexture()

```
void Sketch::createTextTexture ( )  [protected], [inherited]
```

create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

Definition at line 63 of file Sketch.cpp.

```
64 {
65     clearTexture(1);
66     if(text.empty()) return ;
67 
68     SDL_Surface* surface = TTF_RenderText_Solid(font, text.c_str(), fontColor);
69 
70     tes[1] = SDL_CreateTextureFromSurface(ren, surface);
71 
72     coor[1].w = surface->w;
73     coor[1].h = surface->h;
74 
75     crop = coor[1];
76     crop.x = 0;
77     crop.y = 0;
78 
79     if(coor[1].w > coor[0].w || coor[1].h > coor[0].h)
80     {
81         crop = SDL_Rect({
82                 std::max(0, coor[1].w - coor[0].w),
83                 std::max(0, coor[1].h - coor[0].h),
84                 coor[0].w,
85                 coor[0].h
86             });
87         coor[1].w = coor[0].w;
88         coor[1].h = coor[0].h;
89     }
90 
91     align();
92 
93     SDL_FreeSurface(surface);
94 }
```

7.1.3.10 Delete()

```
void Button::Delete ( )
```

Definition at line 101 of file Button.cpp.

```
102 {
103     ren = nullptr;
104     //Object::~Object();
105 
106     if(msg != nullptr)
107         delete [] msg;
108 
109     if(!argv.empty())
110     {
111         for(int i = 0; i < argc; i++)
112             delete [] argv[i];
113         argv.clear();
114         argc = 0;
115     }
116 }
```

7.1.3.11 FillWithColor() [1/2]

```
void Sketch::FillWithColor ( )  [inherited]
```

fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

Definition at line 394 of file Sketch.cpp.

```
395 {
396     int w = coor[0].w;
397     int h = coor[0].h;
398     clearTexture(0);
399 
400     SDL_Surface* surf = SDL_CreateRGBSurfaceWithFormat(0, w, h, 32, SDL_PIXELFORMAT_RGBA32);
```

```
401     SDL_SetSurfaceBlendMode(surf, SDL_BLENDMODE_BLEND);
402
403     SDL_FillRect(surf, nullptr, SDL_MapRGBA(surf->format, color.r, color.g, color.b, color.a));
404
405     SDL_Rect borderRect;
406
407     Uint32 c = SDL_MapRGBA(surf->format, borderColor.r, borderColor.g, borderColor.b, borderColor.a);
408     borderRect = SDL_Rect({0, 0, borderWidth, h});
409     SDL_FillRect(surf, &borderRect, c);
410
411     borderRect = SDL_Rect({0, 0, w, borderWidth});
412     SDL_FillRect(surf, &borderRect, c);
413
414     borderRect = SDL_Rect({0, h - borderWidth, w, borderWidth});
415     SDL_FillRect(surf, &borderRect, c);
416
417     borderRect = SDL_Rect({w - borderWidth, 0, borderWidth, h});
418     SDL_FillRect(surf, &borderRect, c);
419
420     tes[0] = SDL_CreateTextureFromSurface(ren, surf);
421
422     SDL_FreeSurface(surf);
423
424 }
```

7.1.3.12 FillWithColor() [2/2]

```
void Sketch::FillWithColor (
            SDL_Color c )  [inherited]
```

fill background color with color C

Parameters

c	SDL_Color, color to fill fill background color with color C

Definition at line 381 of file Sketch.cpp.

```
382 {
383     SDL_Color temp = color;
384     color = c;
385     FillWithColor();
386     color = temp;
387 }
```

7.1.3.13 getAction()

```
std::string Button::getAction ( )
```

Definition at line 22 of file Button.cpp.

```
23 {
24     return action;
25 }
```

7.1.3.14 getCoor()

```
const SDL_Rect & Object::getCoor ( )  [inherited]
```

return coordinate of this object

Definition at line 133 of file Object.cpp.

```
134 {
135     return coor;
136 }
```

7.1.3.15 getDataStructure()

```
std::string Button::getDataStructure ( )
```

Definition at line 123 of file Button.cpp.

```
124 {
125     if(action != "change screen") return "";
126     if((std::string)(msg) != "working.json") return "";
127     if(argc == 0) return "";
128     return argv[0];
129 }
```

7.1.3.16 getNextScreen()

```
char *const & Button::getNextScreen ( )
```

Definition at line 96 of file Button.cpp.

```
97 {
98     return msg;
99 }
```

7.1.3.17 getText()

```
const std::string & Sketch::getText ( )  [inherited]
```

return text

Definition at line 148 of file Sketch.cpp.

```
149 {
150     return text;
151 }
```

7.1.3.18 hide()

```
void Object::hide ( )  [inherited]
```

hide this object

Definition at line 147 of file Object.cpp.

```
148 {
149     visable = false;
150 }
```

7.1.3.19 highlight()

```
void Sketch::highlight ( )  [inherited]
```

hightlight the sketch this function will change color of background to invert color

Definition at line 867 of file Sketch.cpp.

```
868 {
869     color.r = 255 - color.r;
870     color.g = 255 - color.g;
871     color.b = 255 - color.g;
872     if(color.r > 20 && color.g > 20 && color.b > 20)
873     {
874         color.r -= color.r * 0.3;
875         color.g -= color.g * 0.3;
876         color.b -= color.b * 0.3;
877     }
878     FillWithColor();
879 }
```

7.1.3.20 init() [1/4]

```
void Button::init (
            const char * dir,
            const char * name )
```

Definition at line 31 of file Button.cpp.

```
32 {
33     char* fullname = combineName(name, "json");
34     char* link = combineLink(dir, fullname);
35     std::ifstream fin(link);
36 
37     json mem;
38 
39     fin » mem;
40     fin.close();
41 
42     init(mem);
43 
44     delete [] fullname;
45     delete [] link;
46 }
```

7.1.3.21 init() [2/4]

```
void Button::init (
            const char * name )
```

Definition at line 26 of file Button.cpp.

```
27 {
28     init(GLOBAL::AtrbButtons, name);
29 }
```

7.1.3.22 init() [3/4]

```
void Button::init (
            const json & mem )
```

Definition at line 48 of file Button.cpp.

```
49 {
50     Object::init(mem, ren);
51 
52     if(mem.contains("action"))
53     {
54         action = mem["action"].get<std::string>();
55         if(mem.contains("msg"))
56         {
57             std::string s = mem["msg"].get<std::string>();
58             msg = new char[s.size() + 2];
59             strcpy(msg, s.c_str());
60             s.clear();
61         }
62         if(mem.contains("arg"))
63         {
64             argc = mem["arg"].size();
65             argv.resize(argc);
66             for(int i = 0; i < argc; i++)
67             {
68                 std::string s = mem["arg"][i].get<std::string>();
69                 argv[i] = new char[s.size() + 2];
70                 strcpy(argv[i], s.c_str());
71                 s.clear();
72             }
73         }
74     }
75 }
```

7.1.3.23 init() [4/4]

```
void Object::init (
            const json & mem,
            SDL_Renderer *& r )  [inherited]
```

init this object by json file

Parameters

mem	json file
r	renderer

Definition at line 63 of file Object.cpp.

```
64 {
65     ren = r;
66
67     Sketch::init(mem);
68
69     initTextures(mem);
70     initRect(mem);
71     initVisible(mem);
72 }
```

7.1.3.24 initBorder()

```
void Sketch::initBorder (
            const json & mem )  [protected], [inherited]
```

init border from json

if mem is not contain "border" key, do nothing

if in "border" object contain "width" key, set width of border to be mem["border"]["width"]

if in "border" object contain "color" key, set color of border to be mem["border"]["color"]

example of param mem:

{

"border": {

```
"width": 0,

"color": {

  "r": 0,

  "g": 0,

  "b": 0,

  "a": 0

}
```

}

}

Parameters

mem	json, contain border of sketch

Definition at line 667 of file Sketch.cpp.

```
668 {
669     if(!mem.contains("border")) return ;
670     if(mem["border"].contains("width"))
671         borderWidth = mem["border"]["width"];
672
673     if(mem["border"].contains("color"))
674     {
```

```
675         if(mem["border"]["color"].contains("r"))
676         {
677             borderColor.r = mem["border"]["color"]["r"];
678         }
679         if(mem["border"]["color"].contains("g"))
680         {
681             borderColor.g = mem["border"]["color"]["g"];
682         }
683         if(mem["border"]["color"].contains("b"))
684         {
685             borderColor.b = mem["border"]["color"]["b"];
686         }
687         if(mem["border"]["color"].contains("a"))
688         {
689             borderColor.a = mem["border"]["color"]["a"];
690         }
691     }
692 }
```

7.1.3.25 initColor()

```
void Sketch::initColor (
            const json & mem )  [protected], [inherited]
```

init color from json

if mem is not contain "color" key, do nothing

if in "color" object contain "r" key, set r color of sketch to be mem["color"]["r"]

if in "color" object contain "g" key, set g color of sketch to be mem["color"]["g"]

if in "color" object contain "b" key, set b color of sketch to be mem["color"]["b"]

if in "color" object contain "a" key, set a color of sketch to be mem["color"]["a"]

example of param mem:

{

"color": {

```
 "r": 0,
 "g": 0,
 "b": 0,
 "a": 0
```

}

}

Parameters

mem	json, contain color of sketch

Definition at line 512 of file Sketch.cpp.

```
513 {
514     if(mem.contains("color"))
515     {
516         if(mem["color"].contains("r"))
517             color.r = mem["color"]["r"];
518         if(mem["color"].contains("g"))
519             color.g = mem["color"]["g"];
520         if(mem["color"].contains("b"))
521             color.b = mem["color"]["b"];
522         if(mem["color"].contains("a"))
523             color.a = mem["color"]["a"];
524         cache = color;
525     }
526 }
```

7.1.3.26 initFont()

```
void Sketch::initFont (
            const json & mem )  [protected], [inherited]
```

init font from json

if mem is not contain "font" key, do nothing

get font file and combine with GLOBAL::FontsFolder to get full path of font file

source font from that path and source the size of the font

if in "font" object contain "rect" key, get rect text

if in "font" object contain "color" key, get color text

if int "font" object contain "text", set default text of sketch to be mem["font"]["text"]

example of param mem:

{

"font": {

```
 "name": "font.ttf",
 "size": 0,
 "rect": {
      "x": 0,
      "y": 0,
      "w": 0,
      "h": 0
},
"color": {
      "r": 0,
      "g": 0,
      "b": 0,
      "a": 0
},
"text": "text"
```

}

}

Parameters

mem	json, contain font of sketch

Definition at line 584 of file Sketch.cpp.

```
585 {
586     if(!mem.contains("font")) return ;
587     if(mem["font"].contains("name") && mem["font"].contains("size"))
588     {
589         char* name = combineLink(GLOBAL::FontsFolder, mem["font"]["name"].get<std::string>().c_str());
590         if(font != nullptr)
591         {
592             TTF_CloseFont(font);
593             font = nullptr;
594         }
595         font = TTF_OpenFont(name, mem["font"]["size"]);
596     }
597     if(mem["font"].contains("rect"))
598     {
599         if(mem["font"]["rect"].contains("x"))
600             coor[1].x = mem["font"]["rect"]["x"];
601         if(mem["font"]["rect"].contains("y"))
602             coor[1].y = mem["font"]["rect"]["y"];
603         if(mem["font"]["rect"].contains("align X"))
604             textAlignX = mem["font"]["rect"]["align X"];
605         if(mem["font"]["rect"].contains("align Y"))
606             textAlignX = mem["font"]["rect"]["align Y"];
607     }
608     if(mem["font"].contains("color"))
609     {
610         if(mem["font"]["color"].contains("r"))
611         {
612             fontColor.r = mem["font"]["color"]["r"];
613         }
614         if(mem["font"]["color"].contains("g"))
615         {
616             fontColor.g = mem["font"]["color"]["g"];
617         }
618         if(mem["font"]["color"].contains("b"))
619         {
620             fontColor.b = mem["font"]["color"]["b"];
621         }
622         if(mem["font"]["color"].contains("a"))
623         {
624             fontColor.a = mem["font"]["color"]["a"];
625         }
626     }
627     if(mem["font"].contains("text"))
628     {
629         setText(mem["font"]["text"].get<std::string>());
630     }
631 }
```

7.1.3.27 isChosen()

```
bool Button::isChosen (
            int x,
            int y )
```

Definition at line 11 of file Button.cpp.

```
12 {
13     if(!isVisible() || !isLiesInside(x, y))
14     {
15         pickTexure(0);
16         return false;
17     }
18     pickTexure(1);
19     return true;
20 }
```

7.1.3.28 isLieInside()

```
bool Sketch::isLieInside (
            int x,
            int y )  [inherited]
```

determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

Parameters

x	int
y	int

Returns

bool

Definition at line 813 of file Sketch.cpp.

```
814 {
815     if(x < coor[0].x || coor[0].x + coor[0].w <= x) return false;
816     if(y < coor[0].y || coor[0].y + coor[0].y <= y) return false;
817     return true;
818 }
```

7.1.3.29 isLiesInside() [1/3]

```
bool Object::isLiesInside (
            int x,
            int y )  [inherited]
```

return true if a point lies inside this object

Definition at line 258 of file Object.cpp.

```
259 {
260     if(x < coor.x || coor.x + coor.w <= x)
261         return false;
262     if(y < coor.y || coor.y + coor.h <= y)
263         return false;
264     return true;
265 }
```

7.1.3.30 isLiesInside() [2/3]

```
bool Object::isLiesInside (
            int x,
            int y,
            int w,
            int h )  [inherited]
```

return true if a rectangle inside this object

Definition at line 269 of file Object.cpp.

```
270 {
271     if(x < coor.x || coor.x + coor.w <= x + w)
272         return false;
273     if(y < coor.y || coor.y + coor.h <= y + h)
274         return false;
275     return true;
276 }
```

7.1.3.31 isLiesInside() [3/3]

```
bool Object::isLiesInside (
            SDL_Rect rect )  [inherited]
```

return true if a rectangle inside this object

Definition at line 280 of file Object.cpp.

```
281 {
282     return isLiesInside(rect.x, rect.y, rect.w, rect.h);
283 }
```

7.1.3.32 isVisible()

```
bool Object::isVisible ( )  [inherited]
```

return true if this object is visible

Definition at line 250 of file Object.cpp.

```
251 {
252     return visable;
253 }
```

7.1.3.33 moveTo()

```
void Object::moveTo (
            int x,
            int y,
            double time )  [inherited]
```

move this object to (x, y) in time seconds

Definition at line 316 of file Object.cpp.

```
317 {
318     int dx = x - getCoor().x;
319     int dy = y - getCoor().y;
320
321     if(diff(time, 0))
322     {
323         coor.x = x;
324         coor.y = y;
325         return ;
326     }
327
328     double velo;
329     if(abs(dx) < abs(dy))
330         velo = dy / time;
331     else velo = dx / time;
332
333     int loop = std::min(80.0, abs(velo * time));
334
335     time = time / loop;
336
337
338     for(int i = 1; i <= loop; i++)
339     {
340         Uint32 startTime = SDL_GetTicks();
341
342         addX(-dx * (i - 1) / loop);
343         addX(dx * i / loop);
344         addY(-dy * (i - 1) / loop);
345         addY(dy * i / loop);
346         render(true);
```

```
347         Uint32 deltaTime = SDL_GetTicks() - startTime;
348         startTime = SDL_GetTicks();
349
350         if(deltaTime <= time * 1000)
351             SDL_Delay(time * 1000 - deltaTime);
352     }
353     setX(x);
354     setY(y);
355     render(true);
356 }
```

7.1.3.34 pickTexure()

```
void Object::pickTexure (
              int k )  [inherited]
```

choose top texture

Definition at line 231 of file Object.cpp.

```
232 {
233     if(k >= size()) return ;
234     top = k;
235 }
```

7.1.3.35 popChar()

```
void Sketch::popChar ( )  [inherited]
```

erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create

Definition at line 112 of file Sketch.cpp.

```
113 {
114     if(text.empty()) return ;
115     text.pop_back();
116     createTextTexture();
117 }
```

7.1.3.36 render() [1/2]

```
void Button::render ( )
```

Definition at line 82 of file Button.cpp.

```
83 {
84     if(!isVisible()) return ;
85
86     Object::render(0);
87     return ;
88 }
```

7.1.3.37 render() [2/2]

```
void Button::render (
            bool update )
```

Definition at line 89 of file Button.cpp.

```
90 {
91     if(!isVisible()) return ;
92     Object::render(update);
93     return ;
94 }
```

7.1.3.38 setBorder()

```
void Sketch::setBorder (
            int w,
            int r,
            int g,
            int b,
            int a )  [inherited]
```

set border

set border of sketch

Parameters

w	interger, width of border
r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255
a	interger, alpha value of border color, 0 - 255

Definition at line 355 of file Sketch.cpp.

```
356 {
357     borderWidth = w;
358     borderColor.r = r;
359     borderColor.g = g;
360     borderColor.b = b;
361     borderColor.a = a;
362 }
```

7.1.3.39 setBorderColor()

```
void Sketch::setBorderColor (
            int r,
            int g,
            int b )  [inherited]
```

set border color

Parameters

r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255 set border color

Definition at line 370 of file Sketch.cpp.

```
371 {
372     borderColor.r = r;
373     borderColor.g = g;
374     borderColor.b = b;
375 }
```

7.1.3.40 setColor() [1/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b )  [inherited]
```

set background color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set background color to be (r, g, b)

Definition at line 167 of file Sketch.cpp.

```
168 {
169     color.r = r;
170     color.g = g;
171     color.b = b;
172 }
```

7.1.3.41 setColor() [2/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b,
            int a )  [inherited]
```

set background color to be (r, g, b a)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255
a	interger, alpha value, 0 - 255 set background color to be (r, g, b, a)

Definition at line 182 of file Sketch.cpp.

```
183 {
184     color.r = r;
185     color.g = g;
186     color.b = b;
187     color.a = a;
188 }
```

7.1.3.42 setColor() [3/3]

```
void Sketch::setColor (
            SDL_Color c )  [inherited]
```

set background color to be c

Parameters

c	SDL_Color, background color

Definition at line 156 of file Sketch.cpp.

```
157 {
158     color = c;
159 }
```

7.1.3.43 setCoor() [1/2]

```
void Object::setCoor (
            int x,
            int y,
            int w,
            int h )  [inherited]
```

set coordinate of this object

Parameters

x	x coordinate
y	y coordinate
w	width
h	height

Definition at line 80 of file Object.cpp.

```
81 {
82     coor.x = x;
83     coor.y = y;
84     coor.w = w;
85     coor.h = h;
86 }
```

7.1.3.44 setCoor() [2/2]

```
void Object::setCoor (
            SDL_Rect key )  [inherited]
```

set coordinate of this object

Parameters

key	SDL_Rect

Definition at line 91 of file Object.cpp.

```
92 {
93     coor.x = key.x;
94     coor.y = key.y;
95     coor.w = key.w;
96     coor.h = key.h;
97 }
```

7.1.3.45 setDataStructure()

```
void Button::setDataStructure (
            std::string s )
```

Definition at line 131 of file Button.cpp.

```
132 {
133     action = "change screen";
134 
135     if(msg != nullptr) delete msg;
136     for(int i = 0; i < argv.size(); i++) delete [] argv[i];
137     argv.clear();
138 
139     std::string screen = "working.json";
140     msg = new char[screen.size() + 1];
141     for(int i = 0; i < screen.size(); i++)
142         msg[i] = screen[i];
143 
144     argv.resize(1);
145     argc = 1;
146     argv[0] = new char [s.size() + 1];
147 
148     for(int i = 0; i < s.size(); i++)
149         argv[0][i] = s[i];
150 }
```

7.1.3.46 setH()

```
void Object::setH (
            int h )  [inherited]
```

set height of this object

Parameters

h	height

Definition at line 126 of file Object.cpp.

```
127 {
128     coor.h = h;
129 }
```

7.1.3.47 setInCenterX()

```
void Sketch::setInCenterX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the center of the background texture

Definition at line 269 of file Sketch.cpp.

```
270 {
271     int x = coor[0].x;
272     int w = coor[0].w;
273     coor[1].x = x + (w - coor[1].w) / 2;
274 }
```

7.1.3.48 setInCenterY()

```
void Sketch::setInCenterY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the center of the background texture

Definition at line 279 of file Sketch.cpp.

```
280 {
281     int y = coor[0].y;
282     int h = coor[0].h;
283     coor[1].y = y + (h - coor[1].h) / 2;
284 }
```

7.1.3.49 setOnLeftSideX()

```
void Sketch::setOnLeftSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the left side of the background texture

Definition at line 289 of file Sketch.cpp.

```
290 {
291     coor[1].x = coor[0].x;
292 }
```

7.1.3.50 setOnLeftSideY()

```
void Sketch::setOnLeftSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the top side of the background texture

Definition at line 307 of file Sketch.cpp.

```
308 {
309     coor[1].y = coor[0].y;
310 }
```

7.1.3.51 setOnRightSideX()

```
void Sketch::setOnRightSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the right side of the background texture

Definition at line 297 of file Sketch.cpp.

```
298 {
299     int x = coor[0].x;
300     int w = coor[0].w;
301     coor[1].x = x + w - coor[1].w;
302 }
```

7.1.3.52 setOnRightSideY()

```
void Sketch::setOnRightSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the bottom side of the background texture

Definition at line 315 of file Sketch.cpp.

```
316 {
317     int y = coor[0].y;
318     int h = coor[0].h;
319     coor[1].y = y + h - coor[1].h;
320 }
```

7.1.3.53 setRender()

```
void Sketch::setRender (
            SDL_Renderer *& r )  [inherited]
```

set render

Parameters

r	address of SDL_Renderer pointer set render of sketch

Definition at line 339 of file Sketch.cpp.

```
340 {
341     ren = r;
342 }
```

7.1.3.54 setRenderer()

```
void Button::setRenderer (
            SDL_Renderer *const & r )
```

Definition at line 77 of file Button.cpp.

```
78 {
79     ren = r;
80 }
```

7.1.3.55 setText()

```
void Sketch::setText (
            std::string s )  [inherited]
```

set text to be s

Parameters

s	string that will be set to text set text to be s and create new text texture

Definition at line 124 of file Sketch.cpp.

```
125 {
126     text = s;
127     createTextTexture();
128 }
```

7.1.3.56 setTextColor()

```
void Sketch::setTextColor (
            int r,
            int g,
            int b )  [inherited]
```

set text color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set text color to be (r, g, b) and create new text texture

Definition at line 136 of file Sketch.cpp.

```
137 {
138     fontColor.r = r;
139     fontColor.g = g;
140     fontColor.b = b;
141     createTextTexture();
142 }
```

7.1.3.57 setTextures()

```
void Object::setTextures (
            const json & mem )  [inherited]
```

load textures from json file

Definition at line 154 of file Object.cpp.

```
155 {
156     clearTextures();
157
```

```
158     tes.resize(mem["textures"].size());
159
160     char* FolderName = new char [256];
161     strcpy(FolderName, mem["name"].get<std::string>().c_str());
162
163     for(int i = 0 ; i < size(); i++)
164     {
165         const char* fullname = combineName(
166             mem["textures"][i]["name"].get<std::string>().c_str(),
167             mem["textures"][i]["type"].get<std::string>().c_str()
168         );
169         const char* name = combineLink(
170             FolderName,
171             fullname
172         );
173         const char* link = combineLink(
174             GLOBAL::GraphicsFolder,
175             name
176         );
177
178         SDL_Surface* surf;
179
180         std::string type = mem["textures"][i]["type"].get<std::string>();
181         if(type == "bmp")
182             surf = SDL_LoadBMP(link);
183         else if(type == "png" || type == "jpg")
184             surf = IMG_Load(link);
185
186         tes[i] = SDL_CreateTextureFromSurface(ren, surf);
187
188         delete [] link;
189         delete [] name;
190         delete []fullname;
191         SDL_FreeSurface(surf);
192     }
193     delete [] FolderName;
194 }
```

7.1.3.58 setW()

```
void Object::setW (
            int w )  [inherited]
```

set width of this object

Parameters

w	width

Definition at line 118 of file Object.cpp.

```
119 {
120     coor.w = w;
121 }
```

7.1.3.59 setX()

```
void Object::setX (
            int x )  [inherited]
```

set x coordinate of this object

Parameters

x	x coordinate

Definition at line 102 of file Object.cpp.

```
103 {
104     coor.x = x;
105 }
```

7.1.3.60 setY()

```
void Object::setY (
            int y )  [inherited]
```

set y coordinate of this object

Parameters

y	y coordinate

Definition at line 110 of file Object.cpp.

```
111 {
112     coor.y = y;
113 }
```

7.1.3.61 show()

```
void Object::show ( )  [inherited]
```

show this object

Definition at line 140 of file Object.cpp.

```
141 {
142     visable = true;
143 }
```

7.1.3.62 size()

```
int Object::size ( )  [inherited]
```

return number of textures

Definition at line 208 of file Object.cpp.

```
209 {
210     return tes.size();
211 }
```

7.1.3.63 unHighlight()

```
void Sketch::unHighlight ( )  [inherited]
```

unhightlight the sketch this function will change color of background to normal color

Definition at line 884 of file Sketch.cpp.

```
885 {
886     color = cache;
887     FillWithColor();
888 }
```

The documentation for this class was generated from the following files:

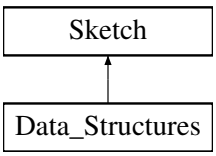
- include/Button.hpp
- src/Button.cpp

7.2 Data_Structures Class Reference

class that handle data structures.

`#include <Data_Structures.hpp>`

Inheritance diagram for Data_Structures:



Public Member Functions

- bool isFinish ()
- void setStep (int k)
- void decStep ()
- int getStep ()
- int size ()

  get number of elements in this data structure
- Data_Structures ()

  constructor of Data_Structures
- ~Data_Structures ()
- void speedUp ()
- void slowDown ()
- void nextStep ()
- void setRender (SDL_Renderer *&r)

  set renderer for this data structure
- void init (const json &mem)

  init data structure from json file
- void loadValue (const json &mem)

  load value from json file
- void render ()

- void render (bool update)
- void create (std::string s)
- void insert (std::string s1, std::string s2, std::mutex &m)
- void erase (std::string s1, std::mutex &m)
- void update (std::string s1, std::string s2, std::mutex &m)
- void search (std::string s2, std::mutex &m)
- void push (std::string s, std::mutex &m)
- void pop (std::string s, std::mutex &m)
- int getType ()
- void custom (std::string s1, std::string s2, std::string s3, std::string s4)
- bool isVisible ()

  get visible this function will return visible of sketch
- void show ()

  show the sketch this function will set visible to true, that will enable the sketch to be rendered
- void hide ()

  hide the sketch this function will set visible to false, that will disable the sketch to be rendered
- void addChar (char ch)

  typing text
- void popChar ()

  erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create
- void setText (std::string s)

  set text to be s
- void setTextColor (int r, int g, int b)

  set text color to be (r, g, b)
- const std::string & getText ()

  return text
- void setColor (SDL_Color c)

  set background color to be c
- void setColor (int r, int g, int b)

  set background color to be (r, g, b)
- void setColor (int r, int g, int b, int a)

  set background color to be (r, g, b a)
- void setCoor (int x, int y, int w, int h)

  set coordinate of sketch
- void setX (int x)

  set coordinate of sketch
- void setY (int y)

  set coordinate of sketch
- void setW (int w)

  set coordinate of sketch
- void setH (int h)

  set coordinate of sketch
- void addX (int x)

  set coordinate of sketch
- void addY (int y)

  set coordinate of sketch
- void setInCenterX ()

  align text texture align x coordinate of text texture to be in the center of the background texture
- void setOnLeftSideX ()

  align text texture align x coordinate of text texture to be in the left side of the background texture
- void setOnRightSideX ()

align text texture align x coordinate of text texture to be in the right side of the background texture

- void setInCenterY ()

  align text texture align y coordinate of text texture to be in the center of the background texture

- void setOnLeftSideY ()

  align text texture align y coordinate of text texture to be in the top side of the background texture

- void setOnRightSideY ()

  align text texture align y coordinate of text texture to be in the bottom side of the background texture

- void align ()

  align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

- SDL_Rect getCoor ()

  get coordinate this function will return coordinate of background of sketch

- void setBorder (int w, int r, int g, int b, int a)

  set border

- void setBorderColor (int r, int g, int b)

  set border color

- void FillWithColor ()

  fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

- void FillWithColor (SDL_Color c)

  fill background color with color C

- void highlight ()

  hightlight the sketch this function will change color of background to invert color

- void unHighlight ()

  unhightlight the sketch this function will change color of background to normal color

- bool isLieInside (int x, int y)

  determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

- void moveTo (int x, int y, double time)

  animation of sketch to move the sketch to point (x, y) in time (second) this function will move the sketch to point (x, y) in time (second)

Protected Member Functions

- void lineDown (int i, int len)
- void lineUp (int i, int len)
- void lineLeft (int i, int len)
- void lineRight (int i, int len)
- void Lining ()
- void Circling (int i, int j)
- void Circling (int i, int j, int k)
- void connect (int i, int j)
- void initStaticArray (const json &mem)
- void StaticArrayCreate (std::string s)
- void StaticArrayInsert (int pos, int value, std::mutex &m)
- void StaticArrayErase (int pos, std::mutex &m)
- void StaticArrayUpdate (int pos, int value, std::mutex &m)
- void StaticArraySearch (int value, std::mutex &m)
- void initDynamicArray (const json &mem)
- void DynamicArrayCreate (std::string s)
- void DynamicArrayInsert (int pos, int value, std::mutex &m)
- void DynamicArrayErase (int pos, std::mutex &m)

- void DynamicArrayUpdate (int pos, int value, std::mutex &m)
- void DynamicArraySearch (int value, std::mutex &m)
- void initSinglyLinkedList (const json &mem)
- void SinglyLinkedListCreate (std::string s)
- void SinglyLinkedListInsert (int pos, int value, std::mutex &m)
- void SinglyLinkedListErase (int pos, std::mutex &m)
- void SinglyLinkedListSearch (int value, std::mutex &m)
- void SinglyLinkedListUpdate (int pos, int value, std::mutex &m)
- void initDoublyLinkedList (const json &mem)
- void DoublyLinkedListCreate (std::string s)
- void DoublyLinkedListInsert (int pos, int value, std::mutex &m)
- void DoublyLinkedListErase (int pos, std::mutex &m)
- void DoublyLinkedListSearch (int value, std::mutex &m)
- void DoublyLinkedListUpdate (int pos, int value, std::mutex &m)
- void initCircularLinkedList (const json &mem)
- void CircularLinkedListCreate (std::string s)
- void CircularLinkedListInsert (int pos, int value, std::mutex &m)
- void CircularLinkedListErase (int pos, std::mutex &m)
- void CircularLinkedListUpdate (int pos, int value, std::mutex &m)
- void CircularLinkedListSearch (int value, std::mutex &m)
- void initStack (const json &mem)
- void StackCreate (std::string s)
- void StackPush (int value, std::mutex &m)
- void StackPop (int value, std::mutex &m)
- void initQueue (const json &mem)
- void QueueCreate (std::string s)
- void QueuePush (int value, std::mutex &m)
- void QueuePop (int value, std::mutex &m)
- void clearTexture (int k)

  clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault
- void initRect (const json &mem)

  set cooridnate of sketch from json
- void initColor (const json &mem)

  init color from json
- void initFont (const json &mem)

  init font from json
- void initBorder (const json &mem)

  init border from json
- void createTextTexture ()

  create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

7.2.1 Detailed Description

class that handle data structures.

Definition at line 17 of file Data_Structures.hpp.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 Data_Structures()

```
Data_Structures::Data_Structures ( )
```

constructor of Data_Structures

Definition at line 13 of file Data_Structures.cpp.

```
14 {
15     finish = true;
16     elements.clear();
17     capacity = 0;
18     ren = nullptr;
19     num = 0;
20     speed = 1;
21     script = nullptr;
22     arrowE = nullptr;
23     depth = 3;
24     lineDepth = 0;
25     circle = false;
26 }
```

7.2.2.2 ~Data_Structures()

```
Data_Structures::~Data_Structures ( )
```

Definition at line 28 of file Data_Structures.cpp.

```
29 {
30     elements.clear();
31     ren = nullptr;
32     capacity = 0;
33     num = 0;
34     if(script != nullptr) delete script;
35     script = nullptr;
36     if(arrowE != nullptr) delete arrowE;
37     arrowE = nullptr;
38     //Sketch::~Sketch();
39 }
```

7.2.3 Member Function Documentation

7.2.3.1 addChar()

```
void Sketch::addChar (
            char ch )  [inherited]
```

typing text

Parameters

ch	character that will be add to the end of the text add a character to the end of the text after that new text texture will be created

Definition at line 101 of file Sketch.cpp.

```
102 {
103     text = text + ch;
```

```
104     createTextTexture();
105 }
```

7.2.3.2 addX()

```
void Sketch::addX (
             int x )  [inherited]
```

set coordinate of sketch

Parameters

x,interger,change	of x coordinate of the top left corner of the sketch add x to x coordinate of sketch

Definition at line 218 of file Sketch.cpp.

```
219 {
220     coor[0].x += x;
221     align();
222 }
```

7.2.3.3 addY()

```
void Sketch::addY (
             int y )  [inherited]
```

set coordinate of sketch

Parameters

y	interger, change of y coordinate of the top left corner of the sketch add y to y coordinate of sketch

Definition at line 229 of file Sketch.cpp.

```
230 {
231     coor[0].y += y;
232     align();
233 }
```

7.2.3.4 align()

```
void Sketch::align ( )  [inherited]
```

align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

Definition at line 761 of file Sketch.cpp.

```
762 {
763
764     if(textAlignX == 1) setOnLeftSideX();
```

```
765     if(textAlignX == 2) setInCenterX();
766     if(textAlignX == 3) setOnRightSideX();
767
768     if(textAlignY == 1) setOnLeftSideY();
769     if(textAlignY == 2) setInCenterY();
770     if(textAlignY == 3) setOnRightSideY();
771 }
```

7.2.3.5 Circling() [1/2]

```
void Data_Structures::Circling (
            int i,
            int j )  [protected]
```

Definition at line 248 of file Data_Structures.cpp.

```
249 {
250     if(i >= num || j >= num || i == j) return ;
251
252     if(i < j)
253     {
254         arrowN->setX(elements[i]->getCoor().x);
255         arrowN->setY(elements[i]->getCoor().y - arrowN->getCoor().h);
256         arrowN->render(false);
257         for(int k = 1; k < depth; k++)
258         {
259             arrowN->addY(-arrowN->getCoor().h);
260             arrowN->render(false);
261         }
262
263         arrowS->setY(arrowN->getCoor().y);
264         arrowS->setX(elements[j]->getCoor().x);
265         arrowS->render(false);
266
267         for(int k = 1; k < depth; k++)
268         {
269             arrowS->addY(arrowS->getCoor().h);
270             arrowS->render(false);
271         }
272
273         arrowE->setX(arrowN->getCoor().x + arrowN->getCoor().w / 2);
274         arrowE->setY(arrowN->getCoor().y - arrowN->getCoor().h - 8);
275
276         do
277         {
278             arrowE->render(false);
279             arrowE->addX(arrowE->getCoor().w);
280         }while(arrowE->getCoor().x - arrowE->getCoor().w < arrowS->getCoor().x);
281         return ;
282     }
283
284     std::swap(i, j);
285
286     arrowS->setX(elements[i]->getCoor().x);
287     arrowS->setY(elements[i]->getCoor().y - arrowN->getCoor().h);
288     arrowS->render(false);
289     for(int k = 1; k < depth; k++)
290     {
291         arrowS->addY(-arrowS->getCoor().h);
292         arrowS->render(false);
293     }
294
295     arrowN->setY(arrowS->getCoor().y);
296     arrowN->setX(elements[j]->getCoor().x);
297     arrowN->render(false);
298
299     for(int k = 1; k < depth; k++)
300     {
301         arrowN->addY(arrowN->getCoor().h);
302         arrowN->render(false);
303     }
304
305     arrowW->setX(arrowS->getCoor().x + arrowS->getCoor().w / 2);
306     arrowW->setY(arrowS->getCoor().y - arrowS->getCoor().h - 8);
307
308     do
309     {
310         arrowW->render(false);
311         arrowW->addX(arrowW->getCoor().w);
```

```
312     }while(arrowW->getCoor().x - arrowW->getCoor().w < arrowN->getCoor().x);
313 }
```

7.2.3.6 Circling() [2/2]

```
void Data_Structures::Circling (
            int i,
            int j,
            int k )  [protected]
```

Definition at line 187 of file Data_Structures.cpp.

```
188 {
189     int temp = depth;
190     depth = k;
191     Circling(i, j);
192     depth = temp;
193 }
```

7.2.3.7 CircularLinkedListCreate()

```
void Data_Structures::CircularLinkedListCreate (
            std::string s )  [protected]
```

Definition at line 1360 of file Data_Structures.cpp.

```
1361 {
1362     SinglyLinkedListCreate(s);
1363     connection[num - 1] = 0;
1364 }
```

7.2.3.8 CircularLinkedListErase()

```
void Data_Structures::CircularLinkedListErase (
            int pos,
            std::mutex & m )  [protected]
```

Definition at line 2440 of file Data_Structures.cpp.

```
2441 {
2442     SinglyLinkedListErase(pos, m);
2443     if(num != 0) connection[num - 1] = 0;
2444 }
```

7.2.3.9 CircularLinkedListInsert()

```
void Data_Structures::CircularLinkedListInsert (
            int pos,
            int value,
            std::mutex & m )  [protected]
```

Definition at line 2434 of file Data_Structures.cpp.

```
2435 {
2436     SinglyLinkedListInsert(pos, value, m);
2437     if(num != 0) connection[num - 1] = 0;
2438 }
```

7.2.3.10 CircularLinkedListSearch()

```
void Data_Structures::CircularLinkedListSearch (
            int value,
            std::mutex & m )  [protected]
```

Definition at line 1966 of file Data_Structures.cpp.

```
1967 {
1968     SinglyLinkedListSearch(value, m);
1969     if(num != 0) connection[num - 1] = 0;
1970 }
```

7.2.3.11 CircularLinkedListUpdate()

```
void Data_Structures::CircularLinkedListUpdate (
            int pos,
            int value,
            std::mutex & m )  [protected]
```

Definition at line 1960 of file Data_Structures.cpp.

```
1961 {
1962     SinglyLinkedListUpdate(pos, value, m);
1963     if(num != 0) connection[num - 1] = 0;
1964 }
```

7.2.3.12 clearTexture()

```
void Sketch::clearTexture (
            int k )  [protected], [inherited]
```

clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault

Parameters

k	integer, index of textures, 0 will be background, 1 will be text if tes[k] is nullptr, do nothing call SDL_DestroyTexture and after that set tes[k] to be nullptr

Definition at line 37 of file Sketch.cpp.

```
38 {
39     if(tes[k] == nullptr) return ;
40     SDL_DestroyTexture(tes[k]);
41     tes[k] = nullptr;
42 }
```

7.2.3.13 connect()

```
void Data_Structures::connect (
            int i,
            int j )  [protected]
```

Definition at line 169 of file Data_Structures.cpp.

```
170 {
171     if(i == -1 || j == -1) return ;
172     if(i == j) return ;
173     if(i + 1 == j && j != capacity)
174     {
175         lineRight(i, 2);
176     }else if(i - 1 == j)
177     {
178         lineLeft(i, 2);
179     }else if (i < capacity && j < capacity)
180         Circling(i, j, depth);
181     else if(i - capacity == j) lineUp(i, 7);
182     else if(j - capacity == i) lineDown(i, 7);
183     else if(i > capacity && j < capacity) connect(i, j + capacity), connect(j + capacity, j);
184     else if(i < capacity && j > capacity) connect(i, i + capacity), connect(i + capacity, j);
185 }
```

7.2.3.14 create()

```
void Data_Structures::create (
            std::string s )
```

Definition at line 1400 of file Data_Structures.cpp.

```
1401 {
1402     finish = false;
1403     if(script != nullptr) script->hide();
1404     if(type == 1) StaticArrayCreate(s);
1405     else if(type == 2) DynamicArrayCreate(s);
1406     else if(type == 3) SinglyLinkedListCreate(s);
1407     else if(type == 4) DoublyLinkedListCreate(s);
1408     else if(type == 5) CircularLinkedListCreate(s);
1409     else if(type == 6) StackCreate(s);
1410     else if(type == 7) QueueCreate(s);
1411     finish = true;
1412 }
```

7.2.3.15 createTextTexture()

```
void Sketch::createTextTexture ( )  [protected], [inherited]
```

create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

Definition at line 63 of file Sketch.cpp.

```
64 {
65     clearTexture(1);
66     if(text.empty()) return ;
67 
68     SDL_Surface* surface = TTF_RenderText_Solid(font, text.c_str(), fontColor);
69 
70     tes[1] = SDL_CreateTextureFromSurface(ren, surface);
71 
72     coor[1].w = surface->w;
73     coor[1].h = surface->h;
74 
75     crop = coor[1];
76     crop.x = 0;
77     crop.y = 0;
78 
79     if(coor[1].w > coor[0].w || coor[1].h > coor[0].h)
80     {
81         crop = SDL_Rect({
82                 std::max(0, coor[1].w - coor[0].w),
83                 std::max(0, coor[1].h - coor[0].h),
84                 coor[0].w,
85                 coor[0].h
86             });
```

```
87          coor[1].w = coor[0].w;
88          coor[1].h = coor[0].h;
89      }
90
91      align();
92
93      SDL_FreeSurface(surface);
94  }
```

7.2.3.16 custom()

```
void Data_Structures::custom (
              std::string s1,
              std::string s2,
              std::string s3,
              std::string s4 )
```

Definition at line 2548 of file Data_Structures.cpp.

```
2549 {
2550     int r, g, b;
2551     getColor(s1, r, g, b);
2552     Sketch::setColor(r, g, b);
2553     Sketch::FillWithColor();
2554
2555     getColor(s2, r, g, b);
2556
2557     for(int i = 0; i < elements.size(); i++)
2558     {
2559         elements[i]->setColor(r, g, b);
2560     }
2561
2562     getColor(s3, r, g, b);
2563
2564     for(int i = 0; i < elements.size(); i++)
2565     {
2566         elements[i]->setTextColor(r, g, b);
2567     }
2568
2569     getColor(s4,r, g, b);
2570
2571     for(int i = 0; i < elements.size(); i++)
2572     {
2573         elements[i]->setBorderColor(r, g, b);
2574         elements[i]->FillWithColor();
2575     }
2576 }
```

7.2.3.17 decStep()

```
void Data_Structures::decStep ( )
```

Definition at line 2361 of file Data_Structures.cpp.

```
2362 {
2363     stepMutex.lock();
2364     step--;
2365     stepMutex.unlock();
2366 }
```

7.2.3.18 DoublyLinkedListCreate()

```
void Data_Structures::DoublyLinkedListCreate (
            std::string s )  [protected]
```

Definition at line 1354 of file Data_Structures.cpp.

```
1355 {
1356     SinglyLinkedListCreate(s);
1357     connection[num - 1] = 0;
1358 }
```

7.2.3.19 DoublyLinkedListErase()

```
void Data_Structures::DoublyLinkedListErase (
            int pos,
            std::mutex & m )  [protected]
```

Definition at line 570 of file Data_Structures.cpp.

```
571 {
572     m.lock();
573     for(int i = 0; i < num; i++)
574         elements[i]->show();
575 
576     json mem;
577     readJson(GLOBAL::AtrbScript + "DoublyLinkedListDelete.json", mem);
578     script->loadObject(mem);
579     script->loadHighlight(mem["highlight"]);
580     script->show();
581 
582     script->highlightLine(0);
583     m.unlock();
584 
585     pos = std::min(pos, num - 1);
586 
587     SDL_Delay(GLOBAL::WAITING / speed);
588     while(getStep() == 0);
589     decStep();
590 
591     m.lock();
592     script->unHighlighLine(0);
593     m.unlock();
594     if(pos == 0)
595     {
596         m.lock();
597         elements[0]->FillWithColor({155, 10, 10, 255});
598         script->highlightLine(2);
599         m.unlock();
600 
601         SDL_Delay(GLOBAL::WAITING / speed);
602         while(getStep() == 0);
603         decStep();
604 
605         m.lock();
606         script->unHighlighLine(2);
607         connection[pos] = -1;
608         script->highlightLine(3);
609         script->highlightLine(4);
610         m.unlock();
611 
612         SDL_Delay(GLOBAL::WAITING / speed);
613         while(getStep() == 0);
614         decStep();
615 
616         m.lock();
617         script->unHighlighLine(3);
618         script->unHighlighLine(4);
619         script->highlightLine(5);
620         elements[0]->FillWithColor({10, 155, 10, 255});
621         for(int i = pos; i + 1 < num; i++)
622         {
623             elements[i]->setText(elements[i + 1]->getText());
624             connection[i] = i + 1;
625         }
626         num--;
```

```
627         elements[num]->hide();
628         m.unlock();
629 
630         SDL_Delay(GLOBAL::WAITING / speed);
631         while(getStep() == 0);
632         decStep();
633 
634         m.lock();
635         elements[0]->FillWithColor();
636         script->unHighlighLine(5);
637         m.unlock();
638         return ;
639     }
640     m.lock();
641     script->highlightLine(8);
642     m.unlock();
643 
644     for(int i = 0; i < pos; i++)
645     {
646         m.lock();
647         script->highlightLine(9);
648         script->highlightLine(10);
649         elements[i]->highlight();
650         m.unlock();
651 
652         SDL_Delay(GLOBAL::WAITING / speed);
653         while(getStep() == 0);
654         decStep();
655 
656         m.lock();
657         script->unHighlighLine(9);
658         script->unHighlighLine(10);
659         elements[i]->unHighlight();
660         m.unlock();
661         SDL_Delay(200 / speed);
662 
663     }
664 
665     SDL_Delay(GLOBAL::WAITING / speed);
666     while(getStep() == 0);
667     decStep();
668 
669 
670     m.lock();
671     elements[pos]->FillWithColor(SDL_Color({155, 10, 10, 255}));
672     script->unHighlighLine(10);
673     script->unHighlighLine(9);
674     script->highlightLine(11);
675     m.unlock();
676 
677     SDL_Delay(GLOBAL::WAITING / speed);
678     while(getStep() == 0);
679     decStep();
680 
681     m.lock();
682     script->unHighlighLine(11);
683     script->highlightLine(12);
684     script->highlightLine(13);
685     elements[pos]->FillWithColor();
686     connection[pos - 1] = pos + 1 != num ?  pos + 1 :  -1;
687     connection[pos] = -1;
688     m.unlock();
689 
690     SDL_Delay(GLOBAL::WAITING / speed);
691     while(getStep() == 0);
692     decStep();
693 
694 
695     m.lock();
696     script->unHighlighLine(12);
697     script->unHighlighLine(13);
698     script->highlightLine(14);
699     if(pos != 0) connection[pos - 1] = pos;
700     for(int i = pos; i + 1 < num; i++)
701     {
702         elements[i]->setText(elements[i + 1]->getText());
703         connection[i] = i + 1;
704     }
705     num--;
706     elements[num]->hide();
707     m.unlock();
708 
709     SDL_Delay(GLOBAL::WAITING / speed);
710     while(getStep() == 0);
711     decStep();
712 
713     m.lock();
```

```
714     script->unHighlighLine(14);
715     m.unlock();
716
717 }
```

7.2.3.20 DoublyLinkedListInsert()

```
void Data_Structures::DoublyLinkedListInsert (
            int pos,
            int value,
            std::mutex & m )  [protected]
```

Definition at line 944 of file Data_Structures.cpp.

```
945 {
946     if(num == capacity) return ;
947     m.lock();
948
949     json mem;
950     readJson(GLOBAL::AtrbScript + "DoublyLinkedListInsert.json", mem);
951     script->loadObject(mem);
952     script->loadHighlight(mem["highlight"]);
953     script->show();
954     script->highlightLine(0);
955
956     for(int i = 0; i < num; i++)
957         elements[i]->show();
958     elements[pos + capacity]->show();
959     elements[pos + capacity]->setText(std::to_string(value));
960     m.unlock();
961
962     SDL_Delay(GLOBAL::WAITING / speed);
963     while(getStep() == 0);
964     decStep();
965
966     m.lock();
967     script->unHighlighLine(0);
968     m.unlock();
969
970
971     if(pos == 0)
972     {
973         m.lock();
974         script->highlightLine(2);
975         script->highlightLine(3);
976         script->highlightLine(4);
977         connection[capacity] = 0;
978         m.unlock();
979
980
981         SDL_Delay(GLOBAL::WAITING / speed);
982         while(getStep() == 0);
983         decStep();
984
985
986         m.lock();
987
988         script->unHighlighLine(2);
989         script->unHighlighLine(3);
990         script->unHighlighLine(4);
991
992         connection[capacity] = -1;
993
994         for(int i = num; i > 0; i--)
995             elements[i]->setText(elements[i - 1]->getText());
996
997         elements[0]->setText(std::to_string(value));
998         elements[num]->show();
999         elements[capacity]->hide();
1000         connection[num - 1] = num;
1001
1002         num++;
1003
1004         m.unlock();
1005
1006         SDL_Delay(GLOBAL::WAITING / speed);
1007         while(getStep() == 0);
1008         decStep();
```

```
1009
1010        return ;
1011    }
1012    m.lock();
1013    script->highlightLine(6);
1014    m.unlock();
1015
1016    SDL_Delay(GLOBAL::WAITING / speed);
1017    while(getStep() == 0);
1018    decStep();
1019
1020
1021    for(int i = 0; i < pos && i < num; i++)
1022    {
1023        m.lock();
1024        elements[i]->highlight();
1025        script->highlightLine(7);
1026        script->highlightLine(8);
1027        m.unlock();
1028
1029        SDL_Delay(GLOBAL::WAITING / speed);
1030        while(getStep() == 0);
1031        decStep();
1032
1033        m.lock();
1034        elements[i]->unHighlight();
1035        script->unHighlighLine(7);
1036        script->unHighlighLine(8);
1037        m.unlock();
1038        SDL_Delay(100 / speed);
1039    }
1040
1041    m.lock();
1042    script->unHighlighLine(6);
1043    elements[pos - 1]->FillWithColor({10, 155, 10, 255});
1044    m.unlock();
1045
1046    SDL_Delay(GLOBAL::WAITING / speed);
1047    while(getStep() == 0);
1048    decStep();
1049
1050    if(pos >= num)
1051    {
1052        m.lock();
1053        script->highlightLine(9);
1054        script->highlightLine(10);
1055        connection[num - 1] = num + capacity;
1056        m.unlock();
1057
1058        SDL_Delay(800 / speed);
1059
1060        m.lock();
1061        connection[num - 1] = num;
1062        elements[pos + capacity]->hide();
1063        connection[num] = -1;
1064        connection[pos + capacity] = -1;
1065        elements[num]->setText(std::to_string(value));
1066        elements[num]->show();
1067        num++;
1068        script->unHighlighLine(9);
1069        script->unHighlighLine(10);
1070        elements[pos - 1]->FillWithColor();
1071        m.unlock();
1072
1073        return ;
1074    }
1075
1076
1077    m.lock();
1078    script->highlightLine(9);
1079    script->highlightLine(10);
1080    connection[pos + capacity] = pos;
1081    m.unlock();
1082
1083    SDL_Delay(GLOBAL::WAITING / speed);
1084    while(getStep() == 0);
1085    decStep();
1086
1087    m.lock();
1088    connection[pos - 1] = pos + capacity;
1089    m.unlock();
1090
1091    SDL_Delay(GLOBAL::WAITING / speed);
1092    while(getStep() == 0);
1093    decStep();
1094
1095    m.lock();
```

```
1096     script->unHighlighLine(9);
1097     script->unHighlighLine(10);
1098     elements[pos - 1]->FillWithColor();
1099     connection[pos - 1] = pos;
1100     connection[pos + capacity] = -1;
1101 
1102     for(int i = num + 1; i > pos; i--)
1103         elements[i]->setText(elements[i - 1]->getText());
1104     elements[pos]->setText(std::to_string(value));
1105     elements[pos + capacity]->hide();
1106     connection[num - 1] = num;
1107     num++;
1108     m.unlock();
1109 }
```

7.2.3.21 DoublyLinkedListSearch()

```
void Data_Structures::DoublyLinkedListSearch (
            int value,
            std::mutex & m )  [protected]
```

Definition at line 1972 of file Data_Structures.cpp.

```
1973 {
1974     SinglyLinkedListSearch(value, m);
1975 }
```

7.2.3.22 DoublyLinkedListUpdate()

```
void Data_Structures::DoublyLinkedListUpdate (
            int pos,
            int value,
            std::mutex & m )  [protected]
```

Definition at line 1955 of file Data_Structures.cpp.

```
1956 {
1957     SinglyLinkedListUpdate(pos, value, m);
1958 }
```

7.2.3.23 DynamicArrayCreate()

```
void Data_Structures::DynamicArrayCreate (
            std::string s )  [protected]
```

Definition at line 382 of file Data_Structures.cpp.

```
383 {
384     int *arr;
385     int n = 0;
386 
387     int ite = 0;
388     num = 0;
389 
390     while(ite < (int)s.size() && num < capacity)
391     {
392         while(ite < (int)s.size() && s[ite] == ' ') ite++;
393         std::string temp;
394         while(ite < (int)s.size() && isdigit(s[ite]))
395             temp += s[ite++];
396         if(temp.empty()) temp = "0";
397         elements[num++]->setText(temp);
```

```
398         ite++;
399     }
400
401     for(int i = num; i < elements.size(); i++)
402         elements[i]->hide();
403
404     for(int i = 0; i < num; i++)
405     {
406         elements[i]->show();
407     }
408 }
```

7.2.3.24 DynamicArrayErase()

```
void Data_Structures::DynamicArrayErase (
            int pos,
            std::mutex & m )  [protected]
```

Definition at line 1800 of file Data_Structures.cpp.

```
1801 {
1802
1803     m.lock();
1804     for(int i = 0; i < num; i++)
1805         elements[i]->show();
1806     json mem;
1807     readJson(GLOBAL::AtrbScript + "DynamicArrayDelete.json", mem);
1808     script->loadObject(mem);
1809     script->loadHighlight(mem["highlight"]);
1810     script->show();
1811     script->highlightLine(0);
1812     m.unlock();
1813
1814     SDL_Delay(GLOBAL::WAITING / speed);
1815     while(getStep() == 0);
1816     decStep();
1817
1818     m.lock();
1819     script->unHighlighLine(0);
1820     script->highlightLine(1);
1821     script->highlightLine(2);
1822
1823     for(int i = 0; i < num - 1; i++)
1824     {
1825         elements[i + capacity]->setText("");
1826         elements[i + capacity]->show();
1827
1828     }
1829     m.unlock();
1830
1831     bool deleted = false;
1832     SDL_Delay(GLOBAL::WAITING / speed);
1833     while(getStep() == 0);
1834     decStep();
1835
1836     m.lock();
1837     script->unHighlighLine(1);
1838     script->unHighlighLine(2);
1839     script->highlightLine(3);
1840     m.unlock();
1841
1842     for(int i = 0; i < num; i++)
1843     {
1844         if(i == pos)
1845         {
1846             deleted = true;
1847
1848             m.lock();
1849             elements[i]->FillWithColor(SDL_Color({175, 20, 20, 255}));
1850             script->highlightLine(5);
1851             m.unlock();
1852
1853
1854             SDL_Delay(GLOBAL::WAITING / speed);
1855             while(getStep() == 0);
1856             decStep();
1857
1858             m.lock();
1859             elements[i]->FillWithColor();
```

```
1860             script->unHighlighLine(5);
1861             m.unlock();
1862 
1863             SDL_Delay(GLOBAL::WAITING / speed);
1864             while(getStep() == 0);
1865             decStep();
1866         }else
1867         {
1868             m.lock();
1869             elements[i]->highlight();
1870             elements[i + capacity - deleted]->highlight();
1871             script->highlightLine(7);
1872             m.unlock();
1873 
1874 
1875             SDL_Delay(GLOBAL::WAITING / speed);
1876             while(getStep() == 0);
1877             decStep();
1878 
1879 
1880             m.lock();
1881             elements[i + capacity - deleted]->setText(elements[i]->getText());
1882             m.unlock();
1883 
1884             SDL_Delay(GLOBAL::WAITING / speed);
1885             while(getStep() == 0);
1886             decStep();
1887 
1888             m.lock();
1889             elements[i]->unHighlight();
1890             elements[i + capacity - deleted]->unHighlight();
1891             script->unHighlighLine(7);
1892 
1893             m.unlock();
1894 
1895             SDL_Delay(100 / speed);
1896             while(getStep() == 0);
1897             decStep();
1898         }
1899     }
1900 
1901     SDL_Delay(200 / speed);
1902     while(getStep() == 0);
1903     decStep();
1904 
1905 
1906     m.lock();
1907     script->unHighlighLine(3);
1908     script->highlightLine(8);
1909     script->highlightLine(9);
1910     num--;
1911     for(int i = 0; i < capacity * 2; i++)
1912         elements[i]->hide();
1913     for(int i = 0; i < num; i++)
1914     {
1915         elements[i]->setText(elements[i + capacity]->getText());
1916         elements[i]->show();
1917     }
1918     m.unlock();
1919 
1920     SDL_Delay(GLOBAL::WAITING / speed);
1921     while(getStep() == 0);
1922     decStep();
1923 
1924     m.lock();
1925 
1926     script->unHighlighLine(8);
1927     script->unHighlighLine(9);
1928     script->highlightLine(10);
1929     m.unlock();
1930 
1931     SDL_Delay(GLOBAL::WAITING / speed);
1932     while(getStep() == 0);
1933     decStep();
1934 
1935     m.lock();
1936     script->unHighlighLine(10);
1937     m.unlock();
1938 }
```

7.2.3.25 DynamicArrayInsert()

```
void Data_Structures::DynamicArrayInsert (
            int pos,
            int value,
            std::mutex & m )  [protected]
```

Definition at line 410 of file Data_Structures.cpp.

```
411 {
412     m.lock();
413     for(int i = 0; i < num + 1; i++)
414         elements[i + capacity]->setText("");
415 
416     json mem;
417     readJson(GLOBAL::AtrbScript + "DynamicArrayInsert.json", mem);
418     script->loadObject(mem);
419     script->loadHighlight(mem["highlight"]);
420     script->show();
421     script->highlightLine(0);
422     m.unlock();
423 
424     SDL_Delay(800 / speed);
425 
426     while(getStep() == 0);
427     decStep();
428 
429     m.lock();
430     script->unHighlighLine(0);
431     script->highlightLine(1);
432     m.unlock();
433 
434     SDL_Delay(GLOBAL::WAITING / speed);
435     while(getStep() == 0);
436     decStep();
437 
438     m.lock();
439     script->unHighlighLine(1);
440     script->highlightLine(3);
441     m.unlock();
442 
443     for(int i = 0; i < num + 1; i++)
444         elements[i + capacity]->show();
445 
446     bool inserted = false;
447 
448     for(int i = 0; i < num; i++)
449     {
450         if(i == pos)
451         {
452             m.lock();
453             elements[i + capacity]->highlight();
454             script->highlightLine(5);
455             script->highlightLine(6);
456             m.unlock();
457 
458             SDL_Delay(GLOBAL::WAITING / speed);
459 
460             while(getStep() == 0);
461             decStep();
462 
463 
464             m.lock();
465             elements[i + capacity]->setText(std::to_string(value));
466             m.unlock();
467 
468             SDL_Delay(500 / speed);
469             while(getStep() == 0);
470             decStep();
471 
472             m.lock();
473             elements[i + capacity]->unHighlight();
474             script->unHighlighLine(5);
475             script->unHighlighLine(6);
476             m.unlock();
477             inserted = true;
478             SDL_Delay(200 / speed);
479 
480         }
481 
482         m.lock();
483         script->highlightLine(8);
484         elements[i]->highlight();
485         elements[i + capacity + inserted]->highlight();
```

```
486         m.unlock();
487
488
489         SDL_Delay(GLOBAL::WAITING / speed);
490         while(getStep() == 0);
491         decStep();
492
493         m.lock();
494         elements[i + capacity + inserted]->setText(elements[i]->getText());
495         m.unlock();
496         SDL_Delay(500 / speed);
497         while(getStep() == 0);
498         decStep();
499
500         m.lock();
501         elements[i]->unHighlight();
502         elements[i + capacity + inserted]->unHighlight();
503         script->unHighlighLine(8);
504         m.unlock();
505     }
506     int i = num;
507     if(i == pos)
508     {
509         m.lock();
510         script->highlightLine(9);
511         script->highlightLine(10);
512         elements[i + capacity]->highlight();
513         m.unlock();
514
515         SDL_Delay(GLOBAL::WAITING / speed);
516
517         while(getStep() == 0);
518         decStep();
519
520
521         m.lock();
522         elements[i + capacity]->setText(std::to_string(value));
523         m.unlock();
524
525         SDL_Delay(500 / speed);
526         while(getStep() == 0);
527         decStep();
528
529         m.lock();
530         script->unHighlighLine(9);
531         script->unHighlighLine(10);
532         elements[i + capacity]->unHighlight();
533         m.unlock();
534         inserted = true;
535         SDL_Delay(200 / speed);
536
537     }
538     while(getStep() == 0);
539     decStep();
540     m.lock();
541     script->highlightLine(11);
542     script->highlightLine(13);
543     m.unlock();
544
545     SDL_Delay(GLOBAL::WAITING / speed);
546     while(getStep() == 0);
547     decStep();
548
549     m.lock();
550     script->unHighlighLine(11);
551     script->unHighlighLine(13);
552     script->highlightLine(14);
553     num++;
554     for(int i = 0; i < num; i++)
555     {
556         elements[i]->setText(elements[i + capacity]->getText());
557         elements[i]->show();
558         elements[i + capacity]->hide();
559     }
560     m.unlock();
561
562     SDL_Delay(GLOBAL::WAITING / speed);
563     while(getStep() == 0);
564     decStep();
565     m.lock();
566     script->unHighlighLine(14);
567     m.unlock();
568 }
```

7.2.3.26 DynamicArraySearch()

```
void Data_Structures::DynamicArraySearch (
            int value,
            std::mutex & m )  [protected]
```

Definition at line 1661 of file Data_Structures.cpp.

```
1662 {
1663
1664     m.lock();
1665     for(int i = 0; i < num; i++) elements[i]->show();
1666
1667     json mem;
1668     readJson(GLOBAL::AtrbScript + "DynamicArraySearch.json", mem);
1669     script->loadObject(mem);
1670     script->loadHighlight(mem["highlight"]);
1671     script->show();
1672     script->highlightLine(0);
1673     m.unlock();
1674
1675     SDL_Delay(GLOBAL::WAITING / speed);
1676     while(getStep() == 0);
1677     decStep();
1678
1679     m.lock();
1680     script->unHighlighLine(0);
1681     script->highlightLine(2);
1682     m.unlock();
1683
1684     SDL_Delay(GLOBAL::WAITING / speed);
1685     while(getStep() == 0);
1686     decStep();
1687
1688     for(int i = 0; i < num; i++)
1689     {
1690         m.lock();
1691         elements[i]->highlight();
1692         m.unlock();
1693
1694         SDL_Delay(GLOBAL::WAITING / speed);
1695         while(getStep() == 0);
1696         decStep();
1697
1698         m.lock();
1699         bool valid = std::to_string(value) == elements[i]->getText();
1700         if(valid)
1701         {
1702             script->highlightLine(4);
1703             script->highlightLine(5);
1704             elements[i]->FillWithColor(SDL_Color({10, 155, 10, 255}));
1705         }else
1706         {
1707             elements[i]->FillWithColor(SDL_Color({155, 10, 10, 255}));
1708         }
1709         m.unlock();
1710
1711         SDL_Delay(GLOBAL::WAITING / speed);
1712         while(getStep() == 0);
1713         decStep();
1714
1715         m.lock();
1716         elements[i]->FillWithColor();
1717         elements[i]->unHighlight();
1718         script->unHighlighLine(4);
1719         script->unHighlighLine(5);
1720         m.unlock();
1721         if(valid) break;
1722     }
1723
1724     SDL_Delay(GLOBAL::WAITING / speed);
1725     while(getStep() == 0);
1726     decStep();
1727
1728     m.lock();
1729     script->highlightLine(6);
1730     m.unlock();
1731
1732     SDL_Delay(GLOBAL::WAITING / speed);
1733     while(getStep() == 0);
1734     decStep();
1735
1736     m.lock();
1737     script->unHighlighLine(6);
1738     m.unlock();
```

```
1739 }
```

7.2.3.27 DynamicArrayUpdate()

```
void Data_Structures::DynamicArrayUpdate (
            int pos,
            int value,
            std::mutex & m )  [protected]
```

Definition at line 1430 of file Data_Structures.cpp.

```
1431 {
1432
1433     m.lock();
1434
1435     for(int i = 0; i < num; i++) elements[i]->show();
1436     elements[pos]->highlight();
1437
1438     json mem;
1439     readJson(GLOBAL::AtrbScript + "DynamicArrayUpdate.json", mem);
1440     script->loadObject(mem);
1441     script->loadHighlight(mem["highlight"]);
1442     script->show();
1443     script->highlightLine(0);
1444     m.unlock();
1445
1446
1447     SDL_Delay(GLOBAL::WAITING / speed);
1448     while(getStep() == 0);
1449     decStep();
1450
1451
1452     m.lock();
1453     script->unHighlighLine(0);
1454     script->highlightLine(1);
1455     elements[pos]->setText(std::to_string(value));
1456     m.unlock();
1457
1458     SDL_Delay(GLOBAL::WAITING / speed);
1459     while(getStep() == 0);
1460     decStep();
1461
1462     m.lock();
1463     elements[pos]->unHighlight();
1464     script->unHighlighLine(1);
1465     script->highlightLine(2);
1466     m.unlock();
1467
1468
1469     SDL_Delay(GLOBAL::WAITING / speed);
1470     while(getStep() == 0);
1471     decStep();
1472
1473     m.lock();
1474     script->unHighlighLine(2);
1475     m.unlock();
1476 }
```

7.2.3.28 erase()

```
void Data_Structures::erase (
            std::string s1,
            std::mutex & m )
```

Definition at line 1940 of file Data_Structures.cpp.

```
1941 {
1942     if(num == 0) return ;
1943     finish = false;
1944     int pos = getFirstInt(s1);
```

```
1945     pos = std::min(pos, num);
1946     step = -1;
1947     if(type == 1) StaticArrayErase(pos, m);
1948     else if(type == 2) DynamicArrayErase(pos, m);
1949     else if(type == 3) SinglyLinkedListErase(pos, m);
1950     else if(type == 4) DoublyLinkedListErase(pos, m);
1951     else if(type == 5) CircularLinkedListErase(pos, m);
1952     finish = true;
1953 }
```

7.2.3.29 FillWithColor() [1/2]

```
void Sketch::FillWithColor ( )  [inherited]
```

fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

Definition at line 394 of file Sketch.cpp.

```
395 {
396     int w = coor[0].w;
397     int h = coor[0].h;
398     clearTexture(0);
399 
400     SDL_Surface* surf = SDL_CreateRGBSurfaceWithFormat(0, w, h, 32, SDL_PIXELFORMAT_RGBA32);
401     SDL_SetSurfaceBlendMode(surf, SDL_BLENDMODE_BLEND);
402 
403     SDL_FillRect(surf, nullptr, SDL_MapRGBA(surf->format, color.r, color.g, color.b, color.a));
404 
405     SDL_Rect borderRect;
406 
407     Uint32 c = SDL_MapRGBA(surf->format, borderColor.r, borderColor.g, borderColor.b, borderColor.a);
408     borderRect = SDL_Rect({0, 0, borderWidth, h});
409     SDL_FillRect(surf, &borderRect, c);
410 
411     borderRect = SDL_Rect({0, 0, w, borderWidth});
412     SDL_FillRect(surf, &borderRect, c);
413 
414     borderRect = SDL_Rect({0, h - borderWidth, w, borderWidth});
415     SDL_FillRect(surf, &borderRect, c);
416 
417     borderRect = SDL_Rect({w - borderWidth, 0, borderWidth, h});
418     SDL_FillRect(surf, &borderRect, c);
419 
420     tes[0] = SDL_CreateTextureFromSurface(ren, surf);
421 
422     SDL_FreeSurface(surf);
423 
424 }
```

7.2.3.30 FillWithColor() [2/2]

```
void Sketch::FillWithColor (
             SDL_Color c )  [inherited]
```

fill background color with color C

Parameters

c	SDL_Color, color to fill fill background color with color C

Definition at line 381 of file Sketch.cpp.

```
382 {
383     SDL_Color temp = color;
384     color = c;
385     FillWithColor();
386     color = temp;
387 }
```

7.2.3.31 getCoor()

```
SDL_Rect Sketch::getCoor ( )  [inherited]
```

get coordinate this function will return coordinate of background of sketch

Returns

SDL_Rect

Definition at line 777 of file Sketch.cpp.

```
778 {
779     return coor[0];
780 }
```

7.2.3.32 getStep()

```
int Data_Structures::getStep ( )
```

Definition at line 2368 of file Data_Structures.cpp.

```
2369 {
2370     int val;
2371     stepMutex.lock();
2372     val = step;
2373     stepMutex.unlock();
2374     return val;
2375 }
```

7.2.3.33 getText()

```
const std::string & Sketch::getText ( )  [inherited]
```

return text

Definition at line 148 of file Sketch.cpp.

```
149 {
150     return text;
151 }
```

7.2.3.34 getType()

```
int Data_Structures::getType ( )
```

Definition at line 2600 of file Data_Structures.cpp.

```
2601 {
2602     return type;
2603 }
```

7.2.3.35 hide()

```
void Sketch::hide ( )  [inherited]
```

hide the sketch this function will set visible to false, that will disable the sketch to be rendered

Definition at line 802 of file Sketch.cpp.

```
803 {
804     visible = false;
805 }
```

7.2.3.36 highlight()

```
void Sketch::highlight ( )  [inherited]
```

hightlight the sketch this function will change color of background to invert color

Definition at line 867 of file Sketch.cpp.

```
868 {
869     color.r = 255 - color.r;
870     color.g = 255 - color.g;
871     color.b = 255 - color.g;
872     if(color.r > 20 && color.g > 20 && color.b > 20)
873     {
874         color.r -= color.r * 0.3;
875         color.g -= color.g * 0.3;
876         color.b -= color.b * 0.3;
877     }
878     FillWithColor();
879 }
```

7.2.3.37 init()

```
void Data_Structures::init (
            const json & mem )
```

init data structure from json file

Definition at line 43 of file Data_Structures.cpp.

```
44 {
45     finish = false;
46     if(!mem.contains("name")) return ;
47     std::string name = mem["name"].get<std::string>();
48     if(mem.contains("script"))
49     {
50         if(script != nullptr) delete script;
51         script = new Script;
52         script->setRender(ren);
53         script->init(mem["script"]);
54     }
55     if(name == "StaticArray.json")
56     {
57         initStaticArray(mem);
58     }else if(name == "DynamicArray.json")
59     {
60         initDynamicArray(mem);
61     }else if(name == "SinglyLinkedList.json")
62     {
63         initSinglyLinkedList(mem);
64     }else if(name == "DoublyLinkedList.json")
65     {
66         initDoublyLinkedList(mem);
67     }else if(name == "CircularLinkedList.json")
68     {
69         initCircularLinkedList(mem);
70     }else if(name == "Stack.json")
71     {
72         initStack(mem);
73     }else if(name == "Queue.json")
74     {
75         initQueue(mem);
76     }
77     finish = true;
78 }
```

7.2.3.38 initBorder()

```
void Sketch::initBorder (
            const json & mem )  [protected], [inherited]
```

init border from json

if mem is not contain "border" key, do nothing

if in "border" object contain "width" key, set width of border to be mem["border"]["width"]

if in "border" object contain "color" key, set color of border to be mem["border"]["color"]

example of param mem:

{

"border": {

```
"width": 0,

"color": {

  "r": 0,

  "g": 0,

  "b": 0,

  "a": 0

}
```

}

}

Parameters

mem	json, contain border of sketch

Definition at line 667 of file Sketch.cpp.

```
668 {
669     if(!mem.contains("border")) return ;
670     if(mem["border"].contains("width"))
671         borderWidth = mem["border"]["width"];
672 
673     if(mem["border"].contains("color"))
674     {
675         if(mem["border"]["color"].contains("r"))
676         {
677             borderColor.r = mem["border"]["color"]["r"];
678         }
679         if(mem["border"]["color"].contains("g"))
680         {
681             borderColor.g = mem["border"]["color"]["g"];
682         }
683         if(mem["border"]["color"].contains("b"))
684         {
685             borderColor.b = mem["border"]["color"]["b"];
686         }
687         if(mem["border"]["color"].contains("a"))
688         {
689             borderColor.a = mem["border"]["color"]["a"];
690         }
691     }
692 }
```

7.2.3.39 initCircularLinkedList()

```
void Data_Structures::initCircularLinkedList (
            const json & mem )  [protected]
```

Definition at line 2446 of file Data_Structures.cpp.

```
2447 {
2448     type = 5;
2449     Sketch::setRender(ren);
2450     Sketch::init(mem);
2451 
2452     elements.clear();
2453 
2454     capacity = 9;
2455     elements.resize(2 * capacity);
2456     connection.resize(2 * capacity, -1);
2457 
2458     if(mem.contains("connect"))
2459     {
```

```
2460         arrowE = new Object;
2461         arrowE->init(mem["connect"][0], ren);
2462 
2463         arrowS = new Object;
2464         arrowS->init(mem["connect"][1], ren);
2465 
2466         arrowW = new Object;
2467         arrowW->init(mem["connect"][2], ren);
2468 
2469         arrowN = new Object;
2470         arrowN->init(mem["connect"][3], ren);
2471     }
2472 
2473     for(int i = 0; i < capacity; i++)
2474     {
2475         elements[i] = new Sketch;
2476         elements[i]->setRender(ren);
2477         connection[i] = (i + 1 != capacity) ?  i + 1 :  -1;
2478 
2479         elements[i + capacity] = new Sketch;
2480         elements[i + capacity]->setRender(ren);
2481 
2482         if(mem.contains("element attributes"))
2483         {
2484             elements[i]->init(mem["element attributes"]);
2485             elements[i + capacity]->init(mem["element attributes"]);
2486 
2487             int dx = mem["element attributes"]["dx"];
2488             int dy = mem["element attributes"]["dy"];
2489 
2490             elements[i]->addX(i * dx);
2491             elements[i + capacity]->addX(i * dx);
2492             elements[i + capacity]->addY(dy);
2493         }
2494     }
2495 }
```

7.2.3.40 initColor()

```
void Sketch::initColor (
            const json & mem )  [protected], [inherited]
```

init color from json

if mem is not contain "color" key, do nothing

if in "color" object contain "r" key, set r color of sketch to be mem["color"]["r"]

if in "color" object contain "g" key, set g color of sketch to be mem["color"]["g"]

if in "color" object contain "b" key, set b color of sketch to be mem["color"]["b"]

if in "color" object contain "a" key, set a color of sketch to be mem["color"]["a"]

example of param mem:

{

"color": {

```
 "r": 0,
 "g": 0,
 "b": 0,
 "a": 0
```

}

}

Parameters

mem	json, contain color of sketch

Definition at line 512 of file Sketch.cpp.

```
513 {
514     if(mem.contains("color"))
515     {
516         if(mem["color"].contains("r"))
517             color.r = mem["color"]["r"];
518         if(mem["color"].contains("g"))
519             color.g = mem["color"]["g"];
520         if(mem["color"].contains("b"))
521             color.b = mem["color"]["b"];
522         if(mem["color"].contains("a"))
523             color.a = mem["color"]["a"];
524         cache = color;
525     }
526 }
```

7.2.3.41 initDoublyLinkedList()

```
void Data_Structures::initDoublyLinkedList (
            const json & mem )  [protected]
```

Definition at line 2383 of file Data_Structures.cpp.

```
2384 {
2385     type = 4;
2386     Sketch::setRender(ren);
2387     Sketch::init(mem);
2388 
2389     elements.clear();
2390 
2391     capacity = 9;
2392     elements.resize(2 * capacity);
2393     connection.resize(2 * capacity, -1);
2394 
2395     if(mem.contains("connect"))
2396     {
2397         arrowE = new Object;
2398         arrowE->init(mem["connect"][0], ren);
2399 
2400         arrowS = new Object;
2401         arrowS->init(mem["connect"][1], ren);
2402 
2403         arrowW = new Object;
2404         arrowW->init(mem["connect"][2], ren);
2405 
2406         arrowN = new Object;
2407         arrowN->init(mem["connect"][3], ren);
2408     }
2409 
2410     for(int i = 0; i < capacity; i++)
2411     {
2412         elements[i] = new Sketch;
2413         elements[i]->setRender(ren);
2414         connection[i] = (i + 1 != capacity) ?  i + 1 :  -1;
2415 
2416         elements[i + capacity] = new Sketch;
2417         elements[i + capacity]->setRender(ren);
2418 
2419         if(mem.contains("element attributes"))
2420         {
2421             elements[i]->init(mem["element attributes"]);
2422             elements[i + capacity]->init(mem["element attributes"]);
2423 
2424             int dx = mem["element attributes"]["dx"];
2425             int dy = mem["element attributes"]["dy"];
2426 
2427             elements[i]->addX(i * dx);
2428             elements[i + capacity]->addX(i * dx);
2429             elements[i + capacity]->addY(dy);
2430         }
2431     }
2432 }
```

7.2.3.42 initDynamicArray()

```
void Data_Structures::initDynamicArray (
            const json & mem )  [protected]
```

Definition at line 135 of file Data_Structures.cpp.

```
136 {
137     type = 2;
138     Sketch::setRender(ren);
139     Sketch::init(mem);
140 
141     capacity = 12;
142     elements.clear();
143     elements.resize(capacity * 2);
144 
145     for(int i = 0; i < capacity; i++)
146     {
147         elements[i] = new Sketch;
148         elements[i]->setRender(ren);
149 
150         elements[i + capacity] = new Sketch;
151         elements[i + capacity]->setRender(ren);
152 
153         if(mem.contains("element attributes"))
154         {
155             elements[i]->init(mem["element attributes"]);
156             elements[i + capacity]->init(mem["element attributes"]);
157             int dx = mem["element attributes"]["dx"];
158             int dy = mem["element attributes"]["dy"];
159 
160             elements[i]->addX(i * dx);
161 
162             elements[i + capacity]->addX(i * dx);
163             elements[i + capacity]->addY(dy);
164         }
165     }
166 
167 }
```

7.2.3.43 initFont()

```
void Sketch::initFont (
            const json & mem )  [protected], [inherited]
```

init font from json

if mem is not contain "font" key, do nothing

get font file and combine with GLOBAL::FontsFolder to get full path of font file

source font from that path and source the size of the font

if in "font" object contain "rect" key, get rect text

if in "font" object contain "color" key, get color text

if int "font" object contain "text", set default text of sketch to be mem["font"]["text"]

example of param mem:

{

"font": {

```
 "name": "font.ttf",

 "size": 0,

 "rect": {

      "x": 0,

      "y": 0,

      "w": 0,

      "h": 0

},

"color": {

      "r": 0,

      "g": 0,

      "b": 0,

      "a": 0

},

"text": "text"


}

}
```

Parameters

mem	json, contain font of sketch

Definition at line 584 of file Sketch.cpp.

```
585 {
586     if(!mem.contains("font")) return ;
587     if(mem["font"].contains("name") && mem["font"].contains("size"))
588     {
589         char* name = combineLink(GLOBAL::FontsFolder, mem["font"]["name"].get<std::string>().c_str());
590         if(font != nullptr)
591         {
592             TTF_CloseFont(font);
593             font = nullptr;
594         }
595         font = TTF_OpenFont(name, mem["font"]["size"]);
596     }
597     if(mem["font"].contains("rect"))
598     {
599         if(mem["font"]["rect"].contains("x"))
600             coor[1].x = mem["font"]["rect"]["x"];
601         if(mem["font"]["rect"].contains("y"))
602             coor[1].y = mem["font"]["rect"]["y"];
603         if(mem["font"]["rect"].contains("align X"))
604             textAlignX = mem["font"]["rect"]["align X"];
605         if(mem["font"]["rect"].contains("align Y"))
606             textAlignX = mem["font"]["rect"]["align Y"];
607     }
608     if(mem["font"].contains("color"))
609     {
610         if(mem["font"]["color"].contains("r"))
611         {
612             fontColor.r = mem["font"]["color"]["r"];
613         }
614         if(mem["font"]["color"].contains("g"))
615         {
616             fontColor.g = mem["font"]["color"]["g"];
617         }
618         if(mem["font"]["color"].contains("b"))
619         {
```

```
620             fontColor.b = mem["font"]["color"]["b"];
621         }
622         if(mem["font"]["color"].contains("a"))
623         {
624             fontColor.a = mem["font"]["color"]["a"];
625         }
626     }
627     if(mem["font"].contains("text"))
628     {
629         setText(mem["font"]["text"].get<std::string>());
630     }
631 }
```

7.2.3.44 initQueue()

```
void Data_Structures::initQueue (
            const json & mem )  [protected]
```

Definition at line 2584 of file Data_Structures.cpp.

```
2585 {
2586     initSinglyLinkedList(mem);
2587     type = 7;
2588 }
```

7.2.3.45 initRect()

```
void Sketch::initRect (
            const json & mem )  [protected], [inherited]
```

set cooridnate of sketch from json

if mem is not contain "rect" key, do nothing

if in "rect" object contain "x" key, set x coordinate of sketch to be mem["rect"]["x"]

if in "rect" object contain "y" key, set y coordinate of sketch to be mem["rect"]["y"]

if in "rect" object contain "w" key, set w coordinate of sketch to be mem["rect"]["w"]

if in "rect" object contain "h" key, set h coordinate of sketch to be mem["rect"]["h"]

example of param mem:

{

```
"rect": {
     "x": 0,
     "y": 0,
     "w": 0,
     "h": 0
}
```

Parameters

mem	json, contain coordinate of sketch

Definition at line 457 of file Sketch.cpp.

```
458 {
459     if(mem.contains("rect"))
460     {
461         if(mem["rect"].contains("x"))
462         {
463             coor[0].x = mem["rect"]["x"];
464         }
465         if(mem["rect"].contains("y"))
466         {
467             coor[0].y = mem["rect"]["y"];
468         }
469         if(mem["rect"].contains("w"))
470         {
471             coor[0].w = mem["rect"]["w"];
472         }
473         if(mem["rect"].contains("h"))
474         {
475             coor[0].h = mem["rect"]["h"];
476         }
477     }
478 }
```

7.2.3.46 initSinglyLinkedList()

```
void Data_Structures::initSinglyLinkedList (
            const json & mem )  [protected]
```

Definition at line 2497 of file Data_Structures.cpp.

```
2498 {
2499     type = 3;
2500     Sketch::setRender(ren);
2501     Sketch::init(mem);
2502 
2503     elements.clear();
2504 
2505     capacity = 9;
2506     elements.resize(2 * capacity);
2507     connection.resize(2 * capacity, -1);
2508 
2509     if(mem.contains("connect"))
2510     {
2511         arrowE = new Object;
2512         arrowE->init(mem["connect"][0], ren);
2513 
2514         arrowS = new Object;
2515         arrowS->init(mem["connect"][1], ren);
2516 
2517         arrowW = new Object;
2518         arrowW->init(mem["connect"][2], ren);
2519 
2520         arrowN = new Object;
2521         arrowN->init(mem["connect"][3], ren);
2522     }
2523 
2524     for(int i = 0; i < capacity; i++)
2525     {
2526         elements[i] = new Sketch;
2527         elements[i]->setRender(ren);
2528         connection[i] = (i + 1 != capacity) ?  i + 1 :  -1;
2529 
2530         elements[i + capacity] = new Sketch;
2531         elements[i + capacity]->setRender(ren);
2532 
2533         if(mem.contains("element attributes"))
2534         {
2535             elements[i]->init(mem["element attributes"]);
2536             elements[i + capacity]->init(mem["element attributes"]);
2537 
2538             int dx = mem["element attributes"]["dx"];
```

```
2539             int dy = mem["element attributes"]["dy"];
2540
2541             elements[i]->addX(i * dx);
2542             elements[i + capacity]->addX(i * dx);
2543             elements[i + capacity]->addY(dy);
2544         }
2545     }
2546 }
```

7.2.3.47 initStack()

```
void Data_Structures::initStack (
            const json & mem )  [protected]
```

Definition at line 2578 of file Data_Structures.cpp.

```
2579 {
2580     initSinglyLinkedList(mem);
2581     type = 6;
2582 }
```

7.2.3.48 initStaticArray()

```
void Data_Structures::initStaticArray (
            const json & mem )  [protected]
```

Definition at line 108 of file Data_Structures.cpp.

```
109 {
110     type = 1;
111     Sketch::setRender(ren);
112     Sketch::init(mem);
113
114     elements.clear();
115
116     capacity = 12;
117     elements.resize(capacity);
118
119     for(int i = 0; i < capacity; i++)
120     {
121         elements[i] = new Sketch;
122         elements[i]->setRender(ren);
123         if(mem.contains("element attributes"))
124         {
125             elements[i]->init(mem["element attributes"]);
126
127             int dx = mem["element attributes"]["dx"];
128             int dy = mem["element attributes"]["dy"];
129
130             elements[i]->addX(i * dx);
131         }
132     }
133 }
```

7.2.3.49 insert()

```
void Data_Structures::insert (
            std::string s1,
            std::string s2,
            std::mutex & m )
```

Definition at line 1414 of file Data_Structures.cpp.

```
1415 {
1416     if(num == capacity) return ;
1417     finish = false;
1418     int pos = getFirstInt(s1);
1419     int value = getFirstInt(s2);
1420     pos = std::min(pos, num);
1421     step = -1;
1422     if(type == 1) StaticArrayInsert(pos, value, m);
1423     else if(type == 2) DynamicArrayInsert(pos, value, m);
1424     else if(type == 3) SinglyLinkedListInsert(pos, value, m);
1425     else if(type == 4) DoublyLinkedListInsert(pos, value, m);
1426     else if(type == 5) CircularLinkedListInsert(pos, value, m);
1427     finish = true;
1428 }
```

7.2.3.50 isFinish()

```
bool Data_Structures::isFinish ( )
```

Definition at line 2378 of file Data_Structures.cpp.

```
2379 {
2380     return finish;
2381 }
```

7.2.3.51 isLieInside()

```
bool Sketch::isLieInside (
            int x,
            int y )  [inherited]
```

determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

Parameters

x	int
y	int

Returns

bool

Definition at line 813 of file Sketch.cpp.

```
814 {
815     if(x < coor[0].x || coor[0].x + coor[0].w <= x) return false;
816     if(y < coor[0].y || coor[0].y + coor[0].y <= y) return false;
817     return true;
818 }
```

7.2.3.52 isVisible()

```
bool Sketch::isVisible ( )  [inherited]
```

get visible this function will return visible of sketch

Returns

bool

Definition at line 786 of file Sketch.cpp.

```
787 {
788     return visible;
789 }
```

7.2.3.53 lineDown()

```
void Data_Structures::lineDown (
            int i,
            int len )  [protected]
```

Definition at line 234 of file Data_Structures.cpp.

```
235 {
236     if(len == 0) return ;
237     arrowS->setX(elements[i]->getCoor().x);
238     arrowS->setY(elements[i]->getCoor().y + elements[i]->getCoor().h);
239     arrowS->render(false);
240 
241     while(--len)
242     {
243         arrowS->addY(arrowS->getCoor().h);
244         arrowS->render(false);
245     }
246 }
```

7.2.3.54 lineLeft()

```
void Data_Structures::lineLeft (
            int i,
            int len )  [protected]
```

Definition at line 195 of file Data_Structures.cpp.

```
196 {
197     arrowW->setX(elements[i]->getCoor().x + arrowW->getCoor().w);
198     arrowW->setY(elements[i]->getCoor().y);
199 
200 
201     while(len--)
202     {
203         arrowW->addX(-arrowW->getCoor().w);
204         arrowW->render(false);
205     }
206 }
```

7.2.3.55 lineRight()

```
void Data_Structures::lineRight (
            int i,
            int len )  [protected]
```

Definition at line 219 of file Data_Structures.cpp.

```
220 {
221     if(len == 0) return ;
222
223     arrowE->setX(elements[i]->getCoor().x - arrowE->getCoor().w + elements[i]->getCoor().w);;
224     arrowE->setY(elements[i]->getCoor().y);
225     arrowE->render(false);
226
227     while(--len)
228     {
229         arrowE->addX(arrowE->getCoor().w);
230         arrowE->render(false);
231     }
232 }
```

7.2.3.56 lineUp()

```
void Data_Structures::lineUp (
            int i,
            int len )  [protected]
```

Definition at line 208 of file Data_Structures.cpp.

```
209 {
210     arrowN->setX(elements[i]->getCoor().x);
211     arrowN->setY(elements[i]->getCoor().y);
212     while(len--)
213     {
214         arrowN->addY(-arrowN->getCoor().h);
215         arrowN->render(false);
216     }
217 }
```

7.2.3.57 Lining()

```
void Data_Structures::Lining ( )  [protected]
```

Definition at line 315 of file Data_Structures.cpp.

```
316 {
317     if(arrowE == nullptr) return ;
318
319     for(int i = 0; i + 1 < num; i++)
320     {
321         arrowE->setX(elements[i]->getCoor().x + elements[i]->getCoor().w);
322         arrowE->setY(elements[i]->getCoor().y);
323         arrowE->render(false);
324     }
325 }
```

7.2.3.58 loadValue()

```
void Data_Structures::loadValue (
            const json & mem )
```

load value from json file

Definition at line 89 of file Data_Structures.cpp.

```
90 {
91     finish = false;
92     if(!mem.contains("name")) return ;
93     if(mem.contains("elements"))
94     {
95         for(int i = 0; i < mem["elements"].size() && i < capacity; i++)
96         {
97             elements[i]->setText(mem["elements"][i].get<std::string>());
98             if(mem.contains("visible") && mem["visible"])
99                 elements[i]->show();
100         }
101         num = mem["elements"].size();
102         if(mem["name"].get<std::string>() == "CircularLinkedList.json")
103             connection[num - 1] = 0;
104     }
105     finish = true;
106 }
```

7.2.3.59 moveTo()

```
void Sketch::moveTo (
            int x,
            int y,
            double time )  [inherited]
```

animation of sketch to move the sketch to point (x, y) in time (second) this function will move the sketch to point (x, y) in time (second)

Parameters

x	int
y	int
time	double

Definition at line 826 of file Sketch.cpp.

```
827 {
828     int dx = x - getCoor().x;
829     int dy = y - getCoor().y;
830
831     if(diff(time, 0))
832     {
833         setX(x);
834         setY(y);
835         return ;
836     }
837
838     double velo;
839
840     if(abs(dx) < abs(dy))
841         velo = dy / time;
842     else velo = dx / time;
843
844     int loop = std::min(80.0, abs(velo * time));
845
846     time = time / loop;
847
848     for(int i = 0; i <= loop; i++)
```

```
849     {
850         Uint32 startTime = SDL_GetTicks();
851 
852         addX(-dx * (i - 1) / loop);
853         addX(dx * i / loop);
854         addY(-dy * (i - 1) / loop);
855         addY(dy * i / loop);
856         render();
857         Uint32 deltaTime = SDL_GetTicks() - startTime;
858         startTime = SDL_GetTicks();
859         if(deltaTime <= time * 1000)
860             SDL_Delay(time * 1000 - deltaTime);
861     }
862 }
```

7.2.3.60 nextStep()

```
void Data_Structures::nextStep ( )
```

Definition at line 2347 of file Data_Structures.cpp.

```
2348 {
2349     stepMutex.lock();
2350     if(step != -1) step = 1;
2351     stepMutex.unlock();
2352 }
```

7.2.3.61 pop()

```
void Data_Structures::pop (
            std::string s,
            std::mutex & m )
```

Definition at line 2182 of file Data_Structures.cpp.

```
2183 {
2184     if(num == 0) return ;
2185     finish = false;
2186     step = -1;
2187     int repeat = getFirstInt(s);
2188     if(type == 6) StackPop(repeat, m);
2189     else if(type == 7) QueuePop(repeat, m);
2190     finish = true;
2191 }
```

7.2.3.62 popChar()

```
void Sketch::popChar ( )  [inherited]
```

erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create

Definition at line 112 of file Sketch.cpp.

```
113 {
114     if(text.empty()) return ;
115     text.pop_back();
116     createTextTexture();
117 }
```

7.2.3.63 push()

```
void Data_Structures::push (
            std::string s,
            std::mutex & m )
```

Definition at line 2007 of file Data_Structures.cpp.

```
2008 {
2009     if(num == capacity) return ;
2010     finish = false;
2011
2012     int value = getFirstInt(s);
2013     step = -1;
2014     if(type == 6) StackPush(value, m);
2015     else if(type == 7) QueuePush(value, m);
2016     finish = true;
2017 }
```

7.2.3.64 QueueCreate()

```
void Data_Structures::QueueCreate (
            std::string s )  [protected]
```

Definition at line 2595 of file Data_Structures.cpp.

```
2596 {
2597     SinglyLinkedListCreate(s);
2598 }
```

7.2.3.65 QueuePop()

```
void Data_Structures::QueuePop (
            int value,
            std::mutex & m )  [protected]
```

Definition at line 2193 of file Data_Structures.cpp.

```
2194 {
2195     m.lock();
2196     for(int i = 0; i < num; i++) elements[i]->show();
2197     for(int i = num; i < elements.size(); i++)
2198         elements[i]->hide();
2199
2200     json mem;
2201     readJson(GLOBAL::AtrbScript + "QueuePop.json", mem);
2202     script->loadObject(mem);
2203     script->loadHighlight(mem["highlight"]);
2204     script->show();
2205     script->highlightLine(0);
2206     m.unlock();
2207
2208     SDL_Delay(GLOBAL::WAITING / speed);
2209     while(getStep() == 0);
2210     decStep();
2211
2212     m.lock();
2213     script->unHighlighLine(0);
2214     script->highlightLine(1);
2215     m.unlock();
2216
2217     while(value-- && num > 0)
2218     {
2219         m.lock();
2220         script->highlightLine(2);
2221         script->highlightLine(3);
2222         elements[0]->FillWithColor({155, 10, 10, 255});
2223         m.unlock();
```

```
2224
2225         SDL_Delay(GLOBAL::WAITING / speed);
2226         while(getStep() == 0);
2227         decStep();
2228
2229         m.lock();
2230         script->unHighlighLine(2);
2231         script->unHighlighLine(3);
2232         script->highlightLine(4);
2233         elements[1]->FillWithColor({10, 155, 10, 255});
2234         connection[0] = -1;
2235         m.unlock();
2236
2237         SDL_Delay(GLOBAL::WAITING / speed);
2238         while(getStep() == 0);
2239         decStep();
2240
2241         m.lock();
2242         script->unHighlighLine(4);
2243         script->highlightLine(5);
2244         connection[0] = 1;
2245         elements[0]->FillWithColor();
2246         elements[1]->FillWithColor();
2247         for(int i = 0; i + 1 < num; i++)
2248             elements[i]->setText(elements[i + 1]->getText());
2249         elements[num - 1]->hide();
2250         num--;
2251         m.unlock();
2252
2253         SDL_Delay(GLOBAL::WAITING / speed);
2254         while(getStep() == 0);
2255         decStep();
2256
2257         m.lock();
2258         script->unHighlighLine(5);
2259         m.unlock();
2260     }
2261     m.lock();
2262     script->unHighlighLine(1);
2263     m.unlock();
2264 }
```

7.2.3.66 QueuePush()

```
void Data_Structures::QueuePush (
            int value,
            std::mutex & m )  [protected]
```

Definition at line 2099 of file Data_Structures.cpp.

```
2100 {
2101     m.lock();
2102     for(int i = 0; i < num; i++) elements[i]->show();
2103
2104     json mem;
2105     readJson(GLOBAL::AtrbScript + "QueueInsert.json", mem);
2106     script->loadObject(mem);
2107     script->loadHighlight(mem["highlight"]);
2108     script->show();
2109     script->highlightLine(0);
2110     m.unlock();
2111
2112     SDL_Delay(GLOBAL::WAITING / speed);
2113     while(getStep() == 0);
2114     decStep();
2115
2116     m.lock();
2117     script->unHighlighLine(0);
2118     script->highlightLine(1);
2119     script->highlightLine(2);
2120     elements[capacity]->show();
2121     elements[capacity]->setText(std::to_string(value));
2122     m.unlock();
2123
2124     SDL_Delay(GLOBAL::WAITING / speed);
2125     while(getStep() == 0);
2126     decStep();
2127
2128     if(num == 0)
```

```
2129     {
2130         m.lock();
2131         script->unHighlighLine(1);
2132         script->unHighlighLine(2);
2133         script->highlightLine(4);
2134         elements[0]->setText(std::to_string(value));
2135         elements[capacity]->hide();
2136         num++;
2137         m.unlock();
2138 
2139         SDL_Delay(GLOBAL::WAITING / speed);
2140         while(getStep() == 0);
2141         decStep();
2142 
2143         m.lock();
2144         script->unHighlighLine(4);
2145         m.unlock();
2146         return ;
2147     }
2148 
2149     m.lock();
2150     script->unHighlighLine(1);
2151     script->unHighlighLine(2);
2152     script->highlightLine(6);
2153     connection[capacity] = 0;
2154     m.unlock();
2155 
2156     SDL_Delay(GLOBAL::WAITING / speed);
2157     while(getStep() == 0);
2158     decStep();
2159 
2160     m.lock();
2161     script->unHighlighLine(6);
2162     script->highlightLine(7);
2163     connection[capacity] = -1;
2164     num++;
2165     for(int i = num; i > 0; i--)
2166         elements[i]->setText(elements[i - 1]->getText());
2167     elements[0]->setText(std::to_string(value));
2168     connection[num - 2] = num - 1;
2169     elements[capacity]->hide();
2170     m.unlock();
2171 
2172     SDL_Delay(GLOBAL::WAITING / speed);
2173     while(getStep() == 0);
2174     decStep();
2175 
2176     m.lock();
2177     script->unHighlighLine(7);
2178     m.unlock();
2179 }
```

7.2.3.67 render() [1/2]

```
void Data_Structures::render ( )
```

Definition at line 327 of file Data_Structures.cpp.

```
328 {
329     if(!isVisible()) return ;
330     Sketch::render();
331     for(int i = 0; i < num; i++)
332         elements[i]->show();
333     for(int i = 0; i < connection.size(); i++)
334     {
335         if(elements[i]->isVisible() && connection[i] != -1 && elements[connection[i]]->isVisible())
336             connect(i, connection[i]);
337     }
338     for(int i = 0; i < elements.size(); i++)
339     {
340         elements[i]->render();
341     }
342     if(script != nullptr)
343     {
344         script->render();
345     }
346 }
```

7.2.3.68 render() [2/2]

```
void Data_Structures::render (
            bool update )
```

7.2.3.69 search()

```
void Data_Structures::search (
            std::string s2,
            std::mutex & m )
```

Definition at line 1993 of file Data_Structures.cpp.

```
1994 {
1995     if(num == 0) return ;
1996     int value = getFirstInt(s2);
1997     step = -1;
1998     finish = false;
1999     if(type == 1) StaticArraySearch(value, m);
2000     else if(type == 2) DynamicArraySearch(value, m);
2001     else if(type == 3) SinglyLinkedListSearch(value, m);
2002     else if(type == 4) DoublyLinkedListSearch(value, m);
2003     else if(type == 5) CircularLinkedListSearch(value, m);
2004     finish = true;
2005 }
```

7.2.3.70 setBorder()

```
void Sketch::setBorder (
            int w,
            int r,
            int g,
            int b,
            int a )  [inherited]
```

set border

set border of sketch

Parameters

w	interger, width of border
r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255
a	interger, alpha value of border color, 0 - 255

Definition at line 355 of file Sketch.cpp.

```
356 {
357     borderWidth = w;
358     borderColor.r = r;
359     borderColor.g = g;
360     borderColor.b = b;
361     borderColor.a = a;
362 }
```

7.2.3.71 setBorderColor()

```
void Sketch::setBorderColor (
            int r,
            int g,
            int b )  [inherited]
```

set border color

Parameters

r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255 set border color

Definition at line 370 of file Sketch.cpp.

```
371 {
372     borderColor.r = r;
373     borderColor.g = g;
374     borderColor.b = b;
375 }
```

7.2.3.72 setColor() [1/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b )  [inherited]
```

set background color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set background color to be (r, g, b)

Definition at line 167 of file Sketch.cpp.

```
168 {
169     color.r = r;
170     color.g = g;
171     color.b = b;
172 }
```

7.2.3.73 setColor() [2/3]

```
void Sketch::setColor (
            int r,
```

```
            int g,
            int b,
            int a )  [inherited]
```

set background color to be (r, g, b a)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255
a	interger, alpha value, 0 - 255 set background color to be (r, g, b, a)

Definition at line 182 of file Sketch.cpp.

```
183 {
184     color.r = r;
185     color.g = g;
186     color.b = b;
187     color.a = a;
188 }
```

7.2.3.74 setColor() [3/3]

```
void Sketch::setColor (
            SDL_Color c )  [inherited]
```

set background color to be c

Parameters

c	SDL_Color, background color

Definition at line 156 of file Sketch.cpp.

```
157 {
158     color = c;
159 }
```

7.2.3.75 setCoor()

```
void Sketch::setCoor (
            int x,
            int y,
            int w,
            int h )  [inherited]
```

set coordinate of sketch

Parameters

x	interger, x coordinate of the top left corner of the sketch
y	interger, y coordinate of the top left corner of the sketch
w	interger, width of the sketch
h	interger, height of the sketch set coordinate of sketch

Definition at line 198 of file Sketch.cpp.

```
199 {
200     coor[0] = SDL_Rect({x, y, w, h});
201     align();
202 }
```

7.2.3.76 setH()

```
void Sketch::setH (
            int h )  [inherited]
```

set coordinate of sketch

Parameters

h	interger, height of the sketch set height of sketch

Definition at line 260 of file Sketch.cpp.

```
261 {
262     coor[0].h = h;
263     align();
264 }
```

7.2.3.77 setInCenterX()

```
void Sketch::setInCenterX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the center of the background texture

Definition at line 269 of file Sketch.cpp.

```
270 {
271     int x = coor[0].x;
272     int w = coor[0].w;
273     coor[1].x = x + (w - coor[1].w) / 2;
274 }
```

7.2.3.78 setInCenterY()

```
void Sketch::setInCenterY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the center of the background texture

Definition at line 279 of file Sketch.cpp.

```
280 {
281     int y = coor[0].y;
282     int h = coor[0].h;
283     coor[1].y = y + (h - coor[1].h) / 2;
284 }
```

7.2.3.79 setOnLeftSideX()

```
void Sketch::setOnLeftSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the left side of the background texture

Definition at line 289 of file Sketch.cpp.

```
290 {
291     coor[1].x = coor[0].x;
292 }
```

7.2.3.80 setOnLeftSideY()

```
void Sketch::setOnLeftSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the top side of the background texture

Definition at line 307 of file Sketch.cpp.

```
308 {
309     coor[1].y = coor[0].y;
310 }
```

7.2.3.81 setOnRightSideX()

```
void Sketch::setOnRightSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the right side of the background texture

Definition at line 297 of file Sketch.cpp.

```
298 {
299     int x = coor[0].x;
300     int w = coor[0].w;
301     coor[1].x = x + w - coor[1].w;
302 }
```

7.2.3.82 setOnRightSideY()

```
void Sketch::setOnRightSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the bottom side of the background texture

Definition at line 315 of file Sketch.cpp.

```
316 {
317     int y = coor[0].y;
318     int h = coor[0].h;
319     coor[1].y = y + h - coor[1].h;
320 }
```

7.2.3.83 setRender()

```
void Data_Structures::setRender (
            SDL_Renderer *& r )
```

set renderer for this data structure

Definition at line 82 of file Data_Structures.cpp.

```
83 {
84     ren = r;
85 }
```

7.2.3.84 setStep()

```
void Data_Structures::setStep (
            int k )
```

Definition at line 2354 of file Data_Structures.cpp.

```
2355 {
2356     stepMutex.lock();
2357     step = k;
2358     stepMutex.unlock();
2359 }
```

7.2.3.85 setText()

```
void Sketch::setText (
            std::string s )  [inherited]
```

set text to be s

Parameters

s	string that will be set to text set text to be s and create new text texture

Definition at line 124 of file Sketch.cpp.

```
125 {
126     text = s;
127     createTextTexture();
128 }
```

7.2.3.86 setTextColor()

```
void Sketch::setTextColor (
            int r,
            int g,
            int b )  [inherited]
```

set text color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set text color to be (r, g, b) and create new text texture

Definition at line 136 of file Sketch.cpp.

```
137 {
138     fontColor.r = r;
139     fontColor.g = g;
140     fontColor.b = b;
141     createTextTexture();
142 }
```

7.2.3.87 setW()

```
void Sketch::setW (
            int w )  [inherited]
```

set coordinate of sketch

Parameters

w	interger, width of the sketch set width of sketch

Definition at line 250 of file Sketch.cpp.

```
251 {
252     coor[0].w = w;
253     align();
254 }
```

7.2.3.88 setX()

```
void Sketch::setX (
            int x )  [inherited]
```

set coordinate of sketch

Parameters

x	interger, x coordinate of the top left corner of the sketch set x coordinate of sketch

Definition at line 208 of file Sketch.cpp.

```
209 {
210     coor[0].x = x;
211     align();
212 }
```

7.2.3.89 setY()

```
void Sketch::setY (
            int y )  [inherited]
```

set coordinate of sketch

Parameters

y	interger, y coordinate of the top left corner of the sketch set y coordinate of sketch

Definition at line 240 of file Sketch.cpp.

```
241 {
242     coor[0].y = y;
243     align();
244 }
```

7.2.3.90 show()

```
void Sketch::show ( )  [inherited]
```

show the sketch this function will set visible to true, that will enable the sketch to be rendered

Definition at line 794 of file Sketch.cpp.

```
795 {
796     visible = true;
797 }
```

7.2.3.91 SinglyLinkedListCreate()

```
void Data_Structures::SinglyLinkedListCreate (
            std::string s )  [protected]
```

Definition at line 1366 of file Data_Structures.cpp.

```
1367 {
1368     connection.clear();
1369     connection.resize(capacity * 2, -1);
1370     int *arr;
1371     int n = 0;
1372
1373     int ite = 0;
1374     num = 0;
1375
1376     while(ite < (int)s.size() && num < capacity)
1377     {
1378         while(ite < (int)s.size() && s[ite] == ' ') ite++;
1379         std::string temp;
1380         while(ite < (int)s.size() && isdigit(s[ite]))
1381             temp += s[ite++];
1382         if(temp.empty()) temp = "0";
1383         elements[num++]->setText(temp);
1384         ite++;
1385     }
1386
1387     for(int i = num; i < 2 * capacity; i++)
1388     {
1389         elements[i]->hide();
1390         elements[i]->setText("");
1391     }
1392     for(int i = 0; i < num; i++)
1393     {
1394         elements[i]->show();
1395         connection[i] = (i + 1 == num) ?  -1 :  i + 1;
1396     }
1397 }
```

7.2.3.92 SinglyLinkedListErase()

```
void Data_Structures::SinglyLinkedListErase (
            int pos,
            std::mutex & m )  [protected]
```

Definition at line 719 of file Data_Structures.cpp.

```
720 {
721     m.lock();
722     for(int i = 0; i < num; i++)
723         elements[i]->show();
724 
725     json mem;
726     readJson(GLOBAL::AtrbScript + "SinglyLinkedListDelete.json", mem);
727     script->loadObject(mem);
728     script->loadHighlight(mem["highlight"]);
729     script->show();
730 
731     script->highlightLine(0);
732     m.unlock();
733 
734     pos = std::min(pos, num - 1);
735 
736     SDL_Delay(GLOBAL::WAITING / speed);
737     while(getStep() == 0);
738     decStep();
739 
740     m.lock();
741     script->unHighlighLine(0);
742     m.unlock();
743     if(pos == 0)
744     {
745         m.lock();
746         elements[0]->FillWithColor({155, 10, 10, 255});
747         script->highlightLine(2);
748         m.unlock();
749 
750         SDL_Delay(GLOBAL::WAITING / speed);
751         while(getStep() == 0);
752         decStep();
753 
754         m.lock();
755         script->unHighlighLine(2);
756         connection[pos] = -1;
757         script->highlightLine(3);
758         m.unlock();
759 
760         SDL_Delay(GLOBAL::WAITING / speed);
761         while(getStep() == 0);
762         decStep();
763 
764         m.lock();
765         script->unHighlighLine(3);
766         script->highlightLine(4);
767         elements[0]->FillWithColor({10, 155, 10, 255});
768         for(int i = pos; i + 1 < num; i++)
769         {
770             elements[i]->setText(elements[i + 1]->getText());
771             connection[i] = i + 1;
772         }
773         num--;
774         elements[num]->hide();
775         m.unlock();
776 
777         SDL_Delay(GLOBAL::WAITING / speed);
778         while(getStep() == 0);
779         decStep();
780 
781         m.lock();
782         elements[0]->FillWithColor();
783         script->unHighlighLine(4);
784         m.unlock();
785         return ;
786     }
787     m.lock();
788     script->highlightLine(6);
789     m.unlock();
790 
791     for(int i = 0; i < pos; i++)
792     {
793         m.lock();
794         script->highlightLine(7);
795         script->highlightLine(8);
796         elements[i]->highlight();
```

```
797         m.unlock();
798
799         SDL_Delay(GLOBAL::WAITING / speed);
800         while(getStep() == 0);
801         decStep();
802
803         m.lock();
804         script->unHighlighLine(7);
805         script->unHighlighLine(8);
806         elements[i]->unHighlight();
807         m.unlock();
808         SDL_Delay(200 / speed);
809
810     }
811
812     SDL_Delay(GLOBAL::WAITING / speed);
813     while(getStep() == 0);
814     decStep();
815
816
817     m.lock();
818     elements[pos]->FillWithColor(SDL_Color({155, 10, 10, 255}));
819     script->unHighlighLine(6);
820     script->highlightLine(9);
821     m.unlock();
822
823     SDL_Delay(GLOBAL::WAITING / speed);
824     while(getStep() == 0);
825     decStep();
826
827     m.lock();
828     script->unHighlighLine(9);
829     script->highlightLine(10);
830     elements[pos]->FillWithColor();
831     connection[pos - 1] = pos + 1 != num ?  pos + 1 :  -1;
832     connection[pos] = -1;
833     m.unlock();
834
835     SDL_Delay(GLOBAL::WAITING / speed);
836     while(getStep() == 0);
837     decStep();
838
839
840     m.lock();
841     script->unHighlighLine(10);
842     script->highlightLine(11);
843     if(pos != 0) connection[pos - 1] = pos;
844     for(int i = pos; i + 1 < num; i++)
845     {
846         elements[i]->setText(elements[i + 1]->getText());
847         connection[i] = i + 1;
848     }
849     num--;
850     elements[num]->hide();
851     m.unlock();
852
853     SDL_Delay(GLOBAL::WAITING / speed);
854     while(getStep() == 0);
855     decStep();
856
857     m.lock();
858     script->unHighlighLine(11);
859     m.unlock();
860 }
```

7.2.3.93 SinglyLinkedListInsert()

```
void Data_Structures::SinglyLinkedListInsert (
              int pos,
              int value,
              std::mutex & m )  [protected]
```

Definition at line 1111 of file Data_Structures.cpp.

```
1112 {
1113     if(num == capacity) return ;
1114     m.lock();
1115
```

```
1116     json mem;
1117     readJson(GLOBAL::AtrbScript + "SinglyLinkedListInsert.json", mem);
1118     script->loadObject(mem);
1119     script->loadHighlight(mem["highlight"]);
1120     script->show();
1121     script->highlightLine(0);
1122 
1123     for(int i = 0; i < num; i++)
1124         elements[i]->show();
1125     elements[pos + capacity]->show();
1126     elements[pos + capacity]->setText(std::to_string(value));
1127     m.unlock();
1128 
1129     SDL_Delay(GLOBAL::WAITING / speed);
1130     while(getStep() == 0);
1131     decStep();
1132 
1133     m.lock();
1134     script->unHighlighLine(0);
1135     m.unlock();
1136 
1137 
1138     if(pos == 0)
1139     {
1140         m.lock();
1141         script->highlightLine(2);
1142         script->highlightLine(3);
1143         script->highlightLine(4);
1144         connection[capacity] = 0;
1145         m.unlock();
1146 
1147 
1148         SDL_Delay(GLOBAL::WAITING / speed);
1149         while(getStep() == 0);
1150         decStep();
1151 
1152 
1153         m.lock();
1154 
1155         script->unHighlighLine(2);
1156         script->unHighlighLine(3);
1157         script->unHighlighLine(4);
1158 
1159         connection[capacity] = -1;
1160 
1161         for(int i = num; i > 0; i--)
1162             elements[i]->setText(elements[i - 1]->getText());
1163 
1164         elements[0]->setText(std::to_string(value));
1165         elements[num]->show();
1166         elements[capacity]->hide();
1167         connection[num - 1] = num;
1168 
1169         num++;
1170 
1171         m.unlock();
1172 
1173         SDL_Delay(GLOBAL::WAITING / speed);
1174         while(getStep() == 0);
1175         decStep();
1176 
1177         return ;
1178     }
1179     m.lock();
1180     script->highlightLine(6);
1181     m.unlock();
1182 
1183     SDL_Delay(GLOBAL::WAITING / speed);
1184     while(getStep() == 0);
1185     decStep();
1186 
1187 
1188     for(int i = 0; i < pos && i < num; i++)
1189     {
1190         m.lock();
1191         elements[i]->highlight();
1192         script->highlightLine(7);
1193         script->highlightLine(8);
1194         m.unlock();
1195 
1196         SDL_Delay(GLOBAL::WAITING / speed);
1197         while(getStep() == 0);
1198         decStep();
1199 
1200         m.lock();
1201         elements[i]->unHighlight();
1202         script->unHighlighLine(7);
```

```
1203         script->unHighlighLine(8);
1204         m.unlock();
1205         SDL_Delay(100 / speed);
1206     }
1207 
1208     m.lock();
1209     script->unHighlighLine(6);
1210     elements[pos - 1]->FillWithColor({10, 155, 10, 255});
1211     m.unlock();
1212 
1213     SDL_Delay(GLOBAL::WAITING / speed);
1214     while(getStep() == 0);
1215     decStep();
1216 
1217     if(pos >= num)
1218     {
1219         m.lock();
1220         script->highlightLine(9);
1221         connection[num - 1] = num + capacity;
1222         m.unlock();
1223 
1224         SDL_Delay(800 / speed);
1225 
1226         m.lock();
1227         connection[num - 1] = num;
1228         elements[pos + capacity]->hide();
1229         connection[num] = -1;
1230         connection[pos + capacity] = -1;
1231         elements[num]->setText(std::to_string(value));
1232         elements[num]->show();
1233         num++;
1234         script->unHighlighLine(9);
1235         elements[pos - 1]->FillWithColor();
1236         m.unlock();
1237 
1238         return ;
1239     }
1240 
1241 
1242     m.lock();
1243     script->highlightLine(9);
1244     connection[pos + capacity] = pos;
1245     m.unlock();
1246 
1247     SDL_Delay(GLOBAL::WAITING / speed);
1248     while(getStep() == 0);
1249     decStep();
1250 
1251     m.lock();
1252     connection[pos - 1] = pos + capacity;
1253     m.unlock();
1254 
1255     SDL_Delay(GLOBAL::WAITING / speed);
1256     while(getStep() == 0);
1257     decStep();
1258 
1259     m.lock();
1260     script->unHighlighLine(9);
1261     elements[pos - 1]->FillWithColor();
1262     connection[pos - 1] = pos;
1263     connection[pos + capacity] = -1;
1264 
1265     for(int i = num + 1; i > pos; i--)
1266         elements[i]->setText(elements[i - 1]->getText());
1267     elements[pos]->setText(std::to_string(value));
1268     elements[pos + capacity]->hide();
1269     connection[num - 1] = num;
1270     num++;
1271     m.unlock();
1272 }
```

7.2.3.94 SinglyLinkedListSearch()

```
void Data_Structures::SinglyLinkedListSearch (
            int value,
            std::mutex & m )  [protected]
```

Definition at line 862 of file Data_Structures.cpp.

```
863 {
864     m.lock();
865     for(int i = 0; i < num; i++)
866         elements[i]->show();
867     json mem;
868     readJson(GLOBAL::AtrbScript + "SinglyLinkedListSearch.json", mem);
869     script->loadObject(mem);
870     script->loadHighlight(mem["highlight"]);
871     script->show();
872     script->highlightLine(0);
873     m.unlock();
874 
875     SDL_Delay(GLOBAL::WAITING / speed);
876     while(getStep() == 0);
877     decStep();
878 
879     m.lock();
880     script->unHighlighLine(0);
881     script->highlightLine(4);
882     m.unlock();
883 
884     for(int i = 0; i < num; i++)
885     {
886         m.lock();
887         elements[i]->highlight();
888         m.unlock();
889 
890         SDL_Delay(GLOBAL::WAITING / speed);
891         while(getStep() == 0);
892         decStep();
893 
894         m.lock();
895         bool valid = std::to_string(value) == elements[i]->getText();
896         if(valid)
897         {
898             script->highlightLine(6);
899             elements[i]->FillWithColor(SDL_Color({10, 155, 10, 255}));
900         }else
901         {
902             elements[i]->FillWithColor(SDL_Color({155, 10, 10, 255}));
903         }
904         m.unlock();
905 
906         SDL_Delay(GLOBAL::WAITING / speed);
907         while(getStep() == 0);
908         decStep();
909 
910 
911         m.lock();
912         elements[i]->FillWithColor();
913         m.unlock();
914 
915         SDL_Delay(GLOBAL::WAITING / speed);
916         while(getStep() == 0);
917         decStep();
918 
919         m.lock();
920         script->unHighlighLine(6);
921         script->highlightLine(7);
922         script->highlightLine(8);
923         elements[i]->unHighlight();
924         m.unlock();
925 
926         SDL_Delay(GLOBAL::WAITING / speed);
927         while(getStep() == 0);
928         decStep();
929 
930         m.lock();
931         script->unHighlighLine(7);
932         script->unHighlighLine(8);
933         m.unlock();
934         if(valid)
935         {
936             break;
937         }
938     }
939     m.lock();
940     script->unHighlighLine(4);
941     m.unlock();
942 }
```

7.2.3.95 SinglyLinkedListUpdate()

```
void Data_Structures::SinglyLinkedListUpdate (
            int pos,
            int value,
            std::mutex & m )  [protected]
```

Definition at line 1588 of file Data_Structures.cpp.

```
1589 {
1590     m.lock();
1591 
1592     json mem;
1593     readJson(GLOBAL::AtrbScript + "SinglyLinkedListUpdate.json", mem);
1594     script->loadObject(mem);
1595     script->loadHighlight(mem["highlight"]);
1596     script->show();
1597     script->highlightLine(0);
1598 
1599     for(int i = 0; i <  num; i++)
1600         elements[i]->show();
1601     m.unlock();
1602 
1603     SDL_Delay(GLOBAL::WAITING / speed);
1604     while(getStep() == 0);
1605     decStep();
1606 
1607     m.lock();
1608     script->unHighlighLine(0);
1609     m.unlock();
1610 
1611     for(int i = 0; i < pos; i++)
1612     {
1613         m.lock();
1614         elements[i]->highlight();
1615         script->highlightLine(3);
1616         script->highlightLine(4);
1617         m.unlock();
1618 
1619         SDL_Delay(GLOBAL::WAITING / speed);
1620         while(getStep() == 0);
1621         decStep();
1622 
1623 
1624         m.lock();
1625         elements[i]->unHighlight();
1626         script->unHighlighLine(3);
1627         script->unHighlighLine(4);
1628         m.unlock();
1629         SDL_Delay(GLOBAL::WAITING / speed);
1630         while(getStep() == 0);
1631         decStep();
1632 
1633     }
1634 
1635     m.lock();
1636     script->highlightLine(5);
1637     elements[pos]->FillWithColor(SDL_Color{10, 155, 10, 255});
1638     m.unlock();
1639 
1640     SDL_Delay(GLOBAL::WAITING / speed);
1641     while(getStep() == 0);
1642     decStep();
1643 
1644     m.lock();
1645     script->unHighlighLine(5);
1646     script->highlightLine(6);
1647 
1648     elements[pos]->setText(std::to_string(value));
1649     m.unlock();
1650 
1651     SDL_Delay(GLOBAL::WAITING / speed);
1652     while(getStep() == 0);
1653     decStep();
1654 
1655     m.lock();
1656     script->unHighlighLine(6);
1657     elements[pos]->FillWithColor();
1658     m.unlock();
1659 }
```

7.2.3.96 size()

```
int Data_Structures::size ( )
```

get number of elements in this data structure

Definition at line 5 of file Data_Structures.cpp.

```
6 {
7     return num;
8 }
```

7.2.3.97 slowDown()

```
void Data_Structures::slowDown ( )
```

Definition at line 2341 of file Data_Structures.cpp.

```
2342 {
2343     if(diff(speed, 0.25)) return ;
2344     speed -= 0.25;
2345 }
```

7.2.3.98 speedUp()

```
void Data_Structures::speedUp ( )
```

Definition at line 2335 of file Data_Structures.cpp.

```
2336 {
2337     if(diff(speed, 3.0)) return ;
2338     speed += 0.25;
2339 }
```

7.2.3.99 StackCreate()

```
void Data_Structures::StackCreate (
            std::string s )  [protected]
```

Definition at line 2590 of file Data_Structures.cpp.

```
2591 {
2592     SinglyLinkedListCreate(s);
2593 }
```

7.2.3.100 StackPop()

```
void Data_Structures::StackPop (
            int value,
            std::mutex & m )  [protected]
```

Definition at line 2266 of file Data_Structures.cpp.

```
2267 {
2268     m.lock();
2269     for(int i = 0; i < num; i++) elements[i]->show();
2270     for(int i = num; i < elements.size(); i++)
2271         elements[i]->hide();
2272 
2273     json mem;
2274     readJson(GLOBAL::AtrbScript + "StackPop.json", mem);
2275     script->loadObject(mem);
2276     script->loadHighlight(mem["highlight"]);
2277     script->show();
2278     script->highlightLine(0);
2279     m.unlock();
2280 
2281     SDL_Delay(GLOBAL::WAITING / speed);
2282     while(getStep() == 0);
2283     decStep();
2284 
2285     m.lock();
2286     script->unHighlighLine(0);
2287     script->highlightLine(1);
2288     m.unlock();
2289 
2290     while(value-- && num > 0)
2291     {
2292         m.lock();
2293         script->highlightLine(2);
2294         script->highlightLine(3);
2295         elements[num - 1]->FillWithColor({155, 10, 10, 255});
2296         m.unlock();
2297 
2298         SDL_Delay(GLOBAL::WAITING / speed);
2299         while(getStep() == 0);
2300         decStep();
2301 
2302         m.lock();
2303         script->unHighlighLine(2);
2304         script->unHighlighLine(3);
2305         script->highlightLine(4);
2306         if(num >= 2) connection[num - 2] = -1;
2307         m.unlock();
2308 
2309         SDL_Delay(GLOBAL::WAITING / speed);
2310         while(getStep() == 0);
2311         decStep();
2312 
2313         m.lock();
2314         script->unHighlighLine(4);
2315         script->highlightLine(5);
2316         elements[num - 1]->hide();
2317         elements[num - 1]->FillWithColor();
2318         num--;
2319         m.unlock();
2320 
2321         SDL_Delay(GLOBAL::WAITING / speed);
2322         while(getStep() == 0);
2323         decStep();
2324 
2325         m.lock();
2326         script->unHighlighLine(5);
2327         m.unlock();
2328     }
2329     m.lock();
2330     script->unHighlighLine(1);
2331     m.unlock();
2332 }
```

7.2.3.101 StackPush()

```
void Data_Structures::StackPush (
            int value,
            std::mutex & m )  [protected]
```

Definition at line 2019 of file Data_Structures.cpp.

```
2020 {
2021     if(num == capacity) return ;
2022     m.lock();
2023     for(int i = 0; i < num; i++) elements[i]->show();
2024 
2025     json mem;
2026     readJson(GLOBAL::AtrbScript + "StackInsert.json", mem);
2027     script->loadObject(mem);
2028     script->loadHighlight(mem["highlight"]);
2029     script->show();
2030     script->highlightLine(0);
2031     m.unlock();
2032 
2033     SDL_Delay(GLOBAL::WAITING / speed);
2034     while(getStep() == 0);
2035     decStep();
2036 
2037     m.lock();
2038     script->unHighlighLine(0);
2039     script->highlightLine(1);
2040     script->highlightLine(2);
2041     elements[num - 1 + capacity]->show();
2042     elements[num - 1 + capacity]->setText(std::to_string(value));
2043     m.unlock();
2044 
2045     SDL_Delay(GLOBAL::WAITING / speed);
2046     while(getStep() == 0);
2047     decStep();
2048 
2049     if(num == 0)
2050     {
2051         m.lock();
2052         script->unHighlighLine(1);
2053         script->unHighlighLine(2);
2054         script->highlightLine(4);
2055         elements[0]->setText(std::to_string(value));
2056         elements[capacity - 1]->hide();
2057         num++;
2058         m.unlock();
2059 
2060         SDL_Delay(GLOBAL::WAITING / speed);
2061         while(getStep() == 0);
2062         decStep();
2063 
2064         m.lock();
2065         script->unHighlighLine(4);
2066         m.unlock();
2067         return ;
2068     }
2069 
2070     m.lock();
2071     script->unHighlighLine(1);
2072     script->unHighlighLine(2);
2073     script->highlightLine(6);
2074     connection[num - 1] = num - 1 + capacity;
2075     m.unlock();
2076 
2077     SDL_Delay(GLOBAL::WAITING / speed);
2078     while(getStep() == 0);
2079     decStep();
2080 
2081     m.lock();
2082     script->unHighlighLine(6);
2083     script->highlightLine(7);
2084     connection[num - 1] = num;
2085     elements[num]->setText(std::to_string(value));
2086     elements[num - 1 + capacity]->hide();
2087     num++;
2088     m.unlock();
2089 
2090     SDL_Delay(GLOBAL::WAITING / speed);
2091     while(getStep() == 0);
2092     decStep();
2093 
2094     m.lock();
2095     script->unHighlighLine(7);
2096     m.unlock();
2097 }
```

7.2.3.102 StaticArrayCreate()

```
void Data_Structures::StaticArrayCreate (
            std::string s )  [protected]
```

Definition at line 354 of file Data_Structures.cpp.

```
355 {
356     int *arr;
357     int n = 0;
358
359     int ite = 0;
360     num = 0;
361
362     while(ite < (int)s.size() && num < capacity)
363     {
364         while(ite < (int)s.size() && s[ite] == ' ') ite++;
365         std::string temp;
366         while(ite < (int)s.size() && isdigit(s[ite]))
367             temp += s[ite++];
368         if(temp.empty()) temp = "0";
369         elements[num++]->setText(temp);
370         ite++;
371     }
372
373     for(int i = num; i < capacity; i++)
374         elements[i]->setText("");
375
376     for(int i = 0; i < capacity; i++)
377     {
378         elements[i]->show();
379     }
380 }
```

7.2.3.103 StaticArrayErase()

```
void Data_Structures::StaticArrayErase (
            int pos,
            std::mutex & m )  [protected]
```

Definition at line 1741 of file Data_Structures.cpp.

```
1742 {
1743
1744     m.lock();
1745     for(int i = 0; i < num; i++) elements[i]->show();
1746     json mem;
1747     readJson(GLOBAL::AtrbScript + "StaticArrayDelete.json", mem);
1748     script->loadObject(mem);
1749     script->loadHighlight(mem["highlight"]);
1750     script->show();
1751     script->highlightLine(3);
1752     m.unlock();
1753     SDL_Delay(1000);
1754
1755     m.lock();
1756     script->unHighlighLine(3);
1757     script->highlightLine(6);
1758     script->highlightLine(7);
1759     m.unlock();
1760     SDL_Delay(300);
1761     num--;
1762     for(int i = pos ; i < num; i++)
1763     {
1764         m.lock();
1765         elements[i]->highlight();
1766         elements[i + 1]->highlight();
1767         m.unlock();
1768
1769         while(getStep() == 0);
1770         decStep();
1771
1772         SDL_Delay(GLOBAL::WAITING / speed);
1773         m.lock();
1774         elements[i]->setText(elements[i + 1]->getText());
1775         m.unlock();
```

```
1776         SDL_Delay(500 / speed);
1777
1778         while(getStep() == 0);
1779
1780         m.lock();
1781         elements[i]->unHighlight();
1782         elements[i + 1]->unHighlight();
1783         m.unlock();
1784         SDL_Delay(100 / speed);
1785     }
1786
1787     m.lock();
1788     script->unHighlighLine(6);
1789     script->unHighlighLine(7);
1790     script->highlightLine(8);
1791     m.unlock();
1792
1793     SDL_Delay(800 / speed);
1794
1795     m.lock();
1796     script->unHighlighLine(8);
1797     m.unlock();
1798 }
```

7.2.3.104 StaticArrayInsert()

```
void Data_Structures::StaticArrayInsert (
            int pos,
            int value,
            std::mutex & m )  [protected]
```

Definition at line 1274 of file Data_Structures.cpp.

```
1275 {
1276
1277     m.lock();
1278     num++;
1279     for(int i = 0; i < num; i++) elements[i]->show();
1280     json mem;
1281     readJson(GLOBAL::AtrbScript + "StaticArrayInsert.json", mem);
1282     script->loadObject(mem);
1283     script->loadHighlight(mem["highlight"]);
1284     script->show();
1285     script->highlightLine(4);
1286     m.unlock();
1287
1288     SDL_Delay(800 / speed);
1289     m.lock();
1290     script->unHighlighLine(4);
1291     script->highlightLine(8);
1292     script->highlightLine(9);
1293     m.unlock();
1294     for(int i = num - 1; i > pos; i--)
1295     {
1296         m.lock();
1297         elements[i]->highlight();
1298         elements[i - 1]->highlight();
1299         m.unlock();
1300
1301         while(getStep() == 0);
1302         decStep();
1303
1304         SDL_Delay(GLOBAL::WAITING / speed);
1305
1306         m.lock();
1307         elements[i]->setText(elements[i - 1]->getText());
1308         m.unlock();
1309         SDL_Delay(500 / speed);
1310
1311         m.lock();
1312         elements[i]->unHighlight();
1313         elements[i - 1]->unHighlight();
1314         m.unlock();
1315         SDL_Delay(100 / speed);
1316     }
1317
1318     while(getStep() == 0);
1319     decStep();
```

```
1320
1321     m.lock();
1322     script->unHighlighLine(8);
1323     script->unHighlighLine(9);
1324     script->highlightLine(11);
1325     script->highlightLine(13);
1326     elements[pos]->highlight();
1327     m.unlock();
1328
1329     SDL_Delay(500 / speed);
1330
1331     m.lock();
1332     elements[pos]->setText(std::to_string(value));
1333     m.unlock();
1334     SDL_Delay(500 / speed);
1335
1336     m.lock();
1337     elements[pos]->unHighlight();
1338     script->unHighlighLine(11);
1339     script->unHighlighLine(13);
1340     m.unlock();
1341     SDL_Delay(500 / speed);
1342
1343     m.lock();
1344     script->highlightLine(15);
1345     m.unlock();
1346
1347     SDL_Delay(800 / speed);
1348
1349     m.lock();
1350     script->unHighlighLine(15);
1351     m.unlock();
1352
1353 }
```

7.2.3.105 StaticArraySearch()

```
void Data_Structures::StaticArraySearch (
            int value,
            std::mutex & m )  [protected]
```

Definition at line 1478 of file Data_Structures.cpp.

```
1479 {
1480     m.lock();
1481     for(int i = 0; i < num; i++) elements[i]->show();
1482     json mem;
1483     readJson(GLOBAL::AtrbScript + "StaticArraySearch.json", mem);
1484     script->loadObject(mem);
1485     script->loadHighlight(mem["highlight"]);
1486     script->highlightLine(3);
1487     script->show();
1488
1489     m.unlock();
1490     SDL_Delay(800 / speed);
1491
1492     m.lock();
1493     script->unHighlighLine(3);
1494     script->highlightLine(5);
1495     script->highlightLine(6);
1496     m.unlock();
1497
1498     for(int i = 0; i < num; i++)
1499     {
1500         m.lock();
1501         elements[i]->highlight();
1502         m.unlock();
1503
1504         SDL_Delay(GLOBAL::WAITING / speed);
1505
1506         while(getStep() == 0);
1507         decStep();
1508
1509         m.lock();
1510         bool valid = std::to_string(value) == elements[i]->getText();
1511         if(valid)
1512         {
1513             script->highlightLine(7);
1514             elements[i]->FillWithColor(SDL_Color({10, 155, 10, 255}));
```

```
1515         }else
1516         {
1517             elements[i]->FillWithColor(SDL_Color({155, 10, 10, 255}));
1518         }
1519         m.unlock();
1520 
1521         SDL_Delay(GLOBAL::WAITING / speed);
1522 
1523         m.lock();
1524         elements[i]->FillWithColor();
1525         m.unlock();
1526 
1527         SDL_Delay(GLOBAL::WAITING / speed);
1528 
1529         m.lock();
1530         elements[i]->unHighlight();
1531         m.unlock();
1532         SDL_Delay(200 / speed);
1533         if(valid)
1534         {
1535             script->unHighlighLine(7);
1536             break;
1537         }
1538     }
1539     m.lock();
1540     script->unHighlighLine(5);
1541     script->unHighlighLine(6);
1542     script->highlightLine(9);
1543     m.unlock();
1544 
1545     SDL_Delay(800 / speed);
1546 
1547     m.lock();
1548     script->unHighlighLine(9);
1549     m.unlock();
1550 }
```

7.2.3.106 StaticArrayUpdate()

```
void Data_Structures::StaticArrayUpdate (
            int pos,
            int value,
            std::mutex & m )  [protected]
```

Definition at line 1552 of file Data_Structures.cpp.

```
1553 {
1554 
1555     m.lock();
1556     for(int i = 0; i < num; i++) elements[i]->show();
1557     json mem;
1558     readJson(GLOBAL::AtrbScript + "StaticArrayUpdate.json", mem);
1559     script->loadObject(mem);
1560     script->loadHighlight(mem["highlight"]);
1561     script->show();
1562     script->highlightLine(3);
1563     m.unlock();
1564     SDL_Delay(800 / speed);
1565 
1566     m.lock();
1567     elements[pos]->highlight();
1568     script->unHighlighLine(3);
1569     script->highlightLine(5);
1570     m.unlock();
1571     while(getStep() == 0);
1572     decStep();
1573 
1574     SDL_Delay(800 / speed);
1575 
1576     m.lock();
1577     script->unHighlighLine(5);
1578     script->highlightLine(6);
1579     elements[pos]->setText(std::to_string(value));
1580     m.unlock();
1581     SDL_Delay(800 / speed);
1582     while(getStep() == 0);
1583     m.lock();
1584     elements[pos]->unHighlight();
```

```
1585     script->unHighlighLine(6);
1586     m.unlock();
1587 }
```

7.2.3.107 unHighlight()

```
void Sketch::unHighlight ( )  [inherited]
```

unhightlight the sketch this function will change color of background to normal color

Definition at line 884 of file Sketch.cpp.

```
885 {
886     color = cache;
887     FillWithColor();
888 }
```

7.2.3.108 update()

```
void Data_Structures::update (
             std::string s1,
             std::string s2,
             std::mutex & m )
```

Definition at line 1977 of file Data_Structures.cpp.

```
1978 {
1979     if(num == 0) return ;
1980     finish = false;
1981     int pos = getFirstInt(s1);
1982     int value = getFirstInt(s2);
1983     step = -1;
1984     pos = std::min(pos, num - 1);
1985     if(type == 1) StaticArrayUpdate(pos, value, m);
1986     else if(type == 2) DynamicArrayUpdate(pos, value, m);
1987     else if(type == 3) SinglyLinkedListUpdate(pos, value, m);
1988     else if(type == 4) DoublyLinkedListUpdate(pos, value, m);
1989     else if(type == 5) CircularLinkedListUpdate(pos, value, m);
1990     finish = true;
1991 }
```

The documentation for this class was generated from the following files:

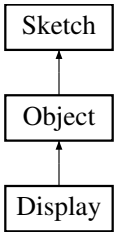
- include/Data_Structures.hpp
- src/Data_Structures.cpp

7.3 Display Class Reference

class that represents a screen. Screen just a rectangle with some buttons on it. window can have many screens

```
#include <Display.hpp>
```

Inheritance diagram for Display:



Public Member Functions

- bool isFreezed ()
- bool isFocus ()

  return true if user are iteractive on this display
- bool changeFocus (int x, int y)

  set this display to be iteractive or not by mouse move
- Display ()

  Constructor of Display.
- void init (const char *dir, const char *name)

  init this display by json file which is in dir/name
- void init (const json &mem)

  init this display by json file
- void loadButtons (const json &mem)

  load buttons from json file
- void loadButton (Button *&but, const json &mem)

  load button from json file
- void setRenderer (SDL_Renderer *const &ren)

  set renderer for this display
- void trigger (int x, int y)

  Run animation when mouse is in trigger area of screen.
- void render ()

  render this display
- void render (bool update)

  render this display
- void mouseMove (int x, int y)

  handle mouse move event
- Button * mousePressedButton (int x, int y)

  get the button that is pressed
- void DeleteButs ()

  delete all buttons
- void hideButton (int k)

  set button visible to false
- void showButton (int k)

  set button visible to true
- void moveTo (int x, int y, double time)

  move the display to (x, y) coordinate in time (second) from the current position
- void appearFromBot (double time)

  move the display to (x, y) coordinate in time (second) from the bottom
- void appearFromRight (double time)

  move the display to (x, y) coordinate in time (second) from the right
- void disappearToBot (double time)

  move the display to bottom coordinate in time (second) from the current position
- void disappearToRight (double time)

  move the display to right coordinate in time (second) from the current position
- int getAppear ()

  return true if the screen have animation
- ~Display ()
- void clearTextures ()

  clear all textures
- void init (const json &mem, SDL_Renderer *&r)

init this object by json file

- void setCoor (int x, int y, int w, int h)

  set coordinate of this object
- void setCoor (SDL_Rect key)

  set coordinate of this object
- void setX (int x)

  set x coordinate of this object
- void setY (int y)

  set y coordinate of this object
- void setW (int w)

  set width of this object
- void setH (int h)

  set height of this object
- const SDL_Rect & getCoor ()

  return coordinate of this object
- bool isLiesInside (int x, int y)

  return true if a point lies inside this object
- bool isLiesInside (int x, int y, int w, int h)

  return true if a rectangle inside this object
- bool isLiesInside (SDL_Rect rect)

  return true if a rectangle inside this object
- void addX (int k)

  add k to x coordinate
- void addY (int k)

  add k to y coordinate
- void addW (int k)

  add k to width
- void addH (int k)

  add k to height
- void show ()

  show this object
- void hide ()

  hide this object
- bool isVisible ()

  return true if this object is visible
- void setTextures (const json &mem)

  load textures from json file
- void pickTexure (int k)

  choose top texture
- int size ()

  return number of textures
- void setRender (SDL_Renderer ∗&r)

  set render
- void addChar (char ch)

  typing text
- void popChar ()

  erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create
- void setText (std::string s)

  set text to be s
- void setTextColor (int r, int g, int b)

set text color to be (r, g, b)

- const std::string & getText ()

  return text

- void setColor (SDL_Color c)

  set background color to be c

- void setColor (int r, int g, int b)

  set background color to be (r, g, b)

- void setColor (int r, int g, int b, int a)

  set background color to be (r, g, b a)

- void setInCenterX ()

  align text texture align x coordinate of text texture to be in the center of the background texture

- void setOnLeftSideX ()

  align text texture align x coordinate of text texture to be in the left side of the background texture

- void setOnRightSideX ()

  align text texture align x coordinate of text texture to be in the right side of the background texture

- void setInCenterY ()

  align text texture align y coordinate of text texture to be in the center of the background texture

- void setOnLeftSideY ()

  align text texture align y coordinate of text texture to be in the top side of the background texture

- void setOnRightSideY ()

  align text texture align y coordinate of text texture to be in the bottom side of the background texture

- void align ()

  align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

- void setBorder (int w, int r, int g, int b, int a)

  set border

- void setBorderColor (int r, int g, int b)

  set border color

- void FillWithColor ()

  fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

- void FillWithColor (SDL_Color c)

  fill background color with color C

- void highlight ()

  hightlight the sketch this function will change color of background to invert color

- void unHighlight ()

  unhightlight the sketch this function will change color of background to normal color

- bool isLieInside (int x, int y)

  determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

Protected Member Functions

- void clearTexture (int k)

  clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault

- void initColor (const json &mem)

  init color from json

- void initFont (const json &mem)

  init font from json

- void initBorder (const json &mem)

  init border from json

- void createTextTexture ()

  create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

7.3.1 Detailed Description

class that represents a screen. Screen just a rectangle with some buttons on it. window can have many screens

Definition at line 19 of file Display.hpp.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 Display()

```
Display::Display ( )
```

Constructor of Display.

Definition at line 5 of file Display.cpp.

```
6 {
7     ren = nullptr;
8     Object::setCoor(0, 0, 960, 540);
9
10     buts.clear();
11     status = 0;
12
13     appear = 0;
14 }
```

7.3.2.2 ~Display()

```
Display::~Display ( )
```

Definition at line 148 of file Display.cpp.

```
149 {
150
151     ren = nullptr;
152     //Object::~Object();
153     DeleteButs();
154 }
```

7.3.3 Member Function Documentation

7.3.3.1 addChar()

```
void Sketch::addChar (
             char ch )  [inherited]
```

typing text

Parameters

ch	character that will be add to the end of the text add a character to the end of the text after that new text texture will be created

Definition at line 101 of file Sketch.cpp.

```
102 {
103     text = text + ch;
104     createTextTexture();
105 }
```

7.3.3.2 addH()

```
void Object::addH (
            int k )  [inherited]
```

add k to height

Definition at line 308 of file Object.cpp.

```
309 {
310     coor.h += k;
311 }
```

7.3.3.3 addW()

```
void Object::addW (
            int k )  [inherited]
```

add k to width

Definition at line 301 of file Object.cpp.

```
302 {
303     coor.w += k;
304 }
```

7.3.3.4 addX()

```
void Object::addX (
            int k )  [inherited]
```

add k to x coordinate

Definition at line 287 of file Object.cpp.

```
288 {
289     coor.x += k;
290 }
```

7.3.3.5 addY()

```
void Object::addY (
            int k )  [inherited]
```

add k to y coordinate

Definition at line 294 of file Object.cpp.

```
295 {
296     coor.y += k;
297 }
```

7.3.3.6 align()

```
void Sketch::align ( )  [inherited]
```

align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

Definition at line 761 of file Sketch.cpp.

```
762 {
763
764     if(textAlignX == 1) setOnLeftSideX();
765     if(textAlignX == 2) setInCenterX();
766     if(textAlignX == 3) setOnRightSideX();
767
768     if(textAlignY == 1) setOnLeftSideY();
769     if(textAlignY == 2) setInCenterY();
770     if(textAlignY == 3) setOnRightSideY();
771 }
```

7.3.3.7 appearFromBot()

```
void Display::appearFromBot (
            double time )
```

move the display to (x, y) coordinate in time (second) from the bottom

Parameters

time	time double

Definition at line 214 of file Display.cpp.

```
215 {
216     int sy = getCoor().y;
217     int dy = 540 - sy;
218
219     for(int i = 0; i < (int) buts.size(); i++)
220         buts[i]->addY(dy);
221
222     setY(540);
223     show();
224     moveTo(getCoor().x, sy, time);
225 }
```

7.3.3.8 appearFromRight()

```
void Display::appearFromRight (
            double time )
```

move the display to (x, y) coordinate in time (second) from the right

Parameters

time	time double

Definition at line 230 of file Display.cpp.

```
231 {
232     int sx = getCoor().x;
233     int dx = 960 - sx;
234 
235     for(int i = 0; i < (int) buts.size(); i++)
236         buts[i]->addX(dx);
237 
238     setX(960);
239     show();
240     moveTo(sx, getCoor().y, time);
241 }
```

7.3.3.9 changeFocus()

```
bool Display::changeFocus (
            int x,
            int y )
```

set this display to be iteractive or not by mouse move

Parameters

x	x coordinate of mouse
y	y coordinate of mouse

Definition at line 27 of file Display.cpp.

```
28 {
29     if(isLiesInside(x, y))
30     {
31         status = 1;
32         return true;
33     }
34     status = 0;
35     return false;
36 }
```

7.3.3.10 clearTexture()

```
void Sketch::clearTexture (
            int k )  [protected], [inherited]
```

clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault

Parameters

k	integer, index of textures, 0 will be background, 1 will be text if tes[k] is nullptr, do nothing call SDL_DestroyTexture and after that set tes[k] to be nullptr

Definition at line 37 of file Sketch.cpp.

```
38 {
39     if(tes[k] == nullptr) return ;
40     SDL_DestroyTexture(tes[k]);
41     tes[k] = nullptr;
42 }
```

7.3.3.11 clearTextures()

```
void Object::clearTextures ( )  [inherited]
```

clear all textures

Definition at line 215 of file Object.cpp.

```
216 {
217     if(!tes.empty())
218     {
219         for(int i = 0; i < size(); i++)
220         {
221             if(tes[i] != nullptr)
222                 SDL_DestroyTexture(tes[i]);
223         }
224         tes.clear();
225     }
226 }
```

7.3.3.12 createTextTexture()

```
void Sketch::createTextTexture ( )  [protected], [inherited]
```

create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

Definition at line 63 of file Sketch.cpp.

```
64 {
65     clearTexture(1);
66     if(text.empty()) return ;
67
68     SDL_Surface* surface = TTF_RenderText_Solid(font, text.c_str(), fontColor);
69
70     tes[1] = SDL_CreateTextureFromSurface(ren, surface);
71
72     coor[1].w = surface->w;
73     coor[1].h = surface->h;
74
75     crop = coor[1];
76     crop.x = 0;
77     crop.y = 0;
78
79     if(coor[1].w > coor[0].w || coor[1].h > coor[0].h)
80     {
81         crop = SDL_Rect({
82                 std::max(0, coor[1].w - coor[0].w),
83                 std::max(0, coor[1].h - coor[0].h),
84                 coor[0].w,
85                 coor[0].h
86             });
87         coor[1].w = coor[0].w;
88         coor[1].h = coor[0].h;
89     }
90
91     align();
92
93     SDL_FreeSurface(surface);
94 }
```

7.3.3.13 DeleteButs()

```
void Display::DeleteButs ( )
```

delete all buttons

Definition at line 120 of file Display.cpp.

```
121 {
122     if(!buts.empty())
123     {
124         for(int i = 0; i < (int) buts.size(); i++)
125             delete buts[i];
126         buts.clear();
127     }
128 }
```

7.3.3.14 disappearToBot()

```
void Display::disappearToBot (
            double time )
```

move the display to bottom coordinate in time (second) from the current position

Parameters

time	time double

Definition at line 247 of file Display.cpp.

```
248 {
249     int sy = getCoor().y;
250     int dy = sy - 540;
251     show();
252     moveTo(getCoor().x, 540, time);
253     hide();
254     setY(sy);
255     for(int i = 0; i < (int) buts.size(); i++)
256         buts[i]->addY(dy);
257 }
```

7.3.3.15 disappearToRight()

```
void Display::disappearToRight (
            double time )
```

move the display to right coordinate in time (second) from the current position

Parameters

time	time double

Definition at line 262 of file Display.cpp.

```
263 {
264     int sx = getCoor().x;
265     int dx = sx - 960;
```

```
266     show();
267     moveTo(960, getCoor().y, time);
268     hide();
269     setX(sx);
270     for(int i = 0; i < (int) buts.size(); i++)
271         buts[i]->addX(dx);
272 }
```

7.3.3.16 FillWithColor() [1/2]

```
void Sketch::FillWithColor ( )  [inherited]
```

fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

Definition at line 394 of file Sketch.cpp.

```
395 {
396     int w = coor[0].w;
397     int h = coor[0].h;
398     clearTexture(0);
399 
400     SDL_Surface* surf = SDL_CreateRGBSurfaceWithFormat(0, w, h, 32, SDL_PIXELFORMAT_RGBA32);
401     SDL_SetSurfaceBlendMode(surf, SDL_BLENDMODE_BLEND);
402 
403     SDL_FillRect(surf, nullptr, SDL_MapRGBA(surf->format, color.r, color.g, color.b, color.a));
404 
405     SDL_Rect borderRect;
406 
407     Uint32 c = SDL_MapRGBA(surf->format, borderColor.r, borderColor.g, borderColor.b, borderColor.a);
408     borderRect = SDL_Rect({0, 0, borderWidth, h});
409     SDL_FillRect(surf, &borderRect, c);
410 
411     borderRect = SDL_Rect({0, 0, w, borderWidth});
412     SDL_FillRect(surf, &borderRect, c);
413 
414     borderRect = SDL_Rect({0, h - borderWidth, w, borderWidth});
415     SDL_FillRect(surf, &borderRect, c);
416 
417     borderRect = SDL_Rect({w - borderWidth, 0, borderWidth, h});
418     SDL_FillRect(surf, &borderRect, c);
419 
420     tes[0] = SDL_CreateTextureFromSurface(ren, surf);
421 
422     SDL_FreeSurface(surf);
423 
424 }
```

7.3.3.17 FillWithColor() [2/2]

```
void Sketch::FillWithColor (
            SDL_Color c )  [inherited]
```

fill background color with color C

Parameters

c	SDL_Color, color to fill fill background color with color C

Definition at line 381 of file Sketch.cpp.

```
382 {
```

```
383     SDL_Color temp = color;
384     color = c;
385     FillWithColor();
386     color = temp;
387 }
```

7.3.3.18 getAppear()

```
int Display::getAppear ( )
```

return true if the screen have animation

Definition at line 276 of file Display.cpp.

```
277 {
278     return appear;
279 }
```

7.3.3.19 getCoor()

```
const SDL_Rect & Object::getCoor ( )  [inherited]
```

return coordinate of this object

Definition at line 133 of file Object.cpp.

```
134 {
135     return coor;
136 }
```

7.3.3.20 getText()

```
const std::string & Sketch::getText ( )  [inherited]
```

return text

Definition at line 148 of file Sketch.cpp.

```
149 {
150     return text;
151 }
```

7.3.3.21 hide()

```
void Object::hide ( )  [inherited]
```

hide this object

Definition at line 147 of file Object.cpp.

```
148 {
149     visable = false;
150 }
```

7.3.3.22 hideButton()

```
void Display::hideButton (
            int k )
```

set button visible to false

Parameters

k	index of button

Definition at line 133 of file Display.cpp.

```
134 {
135     if(k >= (int) buts.size()) return ;
136     buts[k]->hide();
137 }
```

7.3.3.23 highlight()

```
void Sketch::highlight ( )  [inherited]
```

hightlight the sketch this function will change color of background to invert color

Definition at line 867 of file Sketch.cpp.

```
868 {
869     color.r = 255 - color.r;
870     color.g = 255 - color.g;
871     color.b = 255 - color.g;
872     if(color.r > 20 && color.g > 20 && color.b > 20)
873     {
874         color.r -= color.r * 0.3;
875         color.g -= color.g * 0.3;
876         color.b -= color.b * 0.3;
877     }
878     FillWithColor();
879 }
```

7.3.3.24 init() [1/3]

```
void Display::init (
            const char * dir,
            const char * name )
```

init this display by json file which is in dir/name

Parameters

dir	directory of json file
name	name of json file

Definition at line 42 of file Display.cpp.

```
43 {
44     json mem;
45 
46     readjson(dir, name, mem);
47 
48     init(mem[0]);
49 }
```

7.3.3.25 init() [2/3]

```
void Display::init (
            const json & mem )
```

init this display by json file

Parameters

mem	json file init buttons, animation (if it has)

Definition at line 55 of file Display.cpp.

```
56 {
57     Object::init(mem, ren);
58 
59     if(mem.contains("buttons"))
60     {
61         loadButtons(mem["buttons"]);
62     }
63     if(mem.contains("appear from"))
64     {
65         if(mem["appear from"].get<std::string>() == "bottom")
66             appear = 1;
67         else if(mem["appear from"].get<std::string>() == "right")
68             appear = 2;
69     }
70 }
```

7.3.3.26 init() [3/3]

```
void Object::init (
            const json & mem,
            SDL_Renderer *& r )  [inherited]
```

init this object by json file

Parameters

mem	json file
r	renderer

Definition at line 63 of file Object.cpp.

```
64 {
65     ren = r;
66 
67     Sketch::init(mem);
68 
69     initTextures(mem);
70     initRect(mem);
71     initVisible(mem);
72 }
```

7.3.3.27 initBorder()

```
void Sketch::initBorder (
            const json & mem )  [protected], [inherited]
```

init border from json

if mem is not contain "border" key, do nothing

if in "border" object contain "width" key, set width of border to be mem["border"]["width"]

if in "border" object contain "color" key, set color of border to be mem["border"]["color"]

example of param mem:

{

"border": {

```
"width": 0,

"color": {

  "r": 0,

  "g": 0,

  "b": 0,

  "a": 0

}
```

}

}

Parameters

mem	json, contain border of sketch

Definition at line 667 of file Sketch.cpp.

```
668 {
669     if(!mem.contains("border")) return ;
670     if(mem["border"].contains("width"))
671         borderWidth = mem["border"]["width"];
672 
673     if(mem["border"].contains("color"))
674     {
675         if(mem["border"]["color"].contains("r"))
676         {
677             borderColor.r = mem["border"]["color"]["r"];
678         }
679         if(mem["border"]["color"].contains("g"))
680         {
681             borderColor.g = mem["border"]["color"]["g"];
682         }
683         if(mem["border"]["color"].contains("b"))
684         {
685             borderColor.b = mem["border"]["color"]["b"];
686         }
687         if(mem["border"]["color"].contains("a"))
688         {
689             borderColor.a = mem["border"]["color"]["a"];
690         }
691     }
692 }
```

7.3.3.28 initColor()

```
void Sketch::initColor (
            const json & mem )  [protected], [inherited]
```

init color from json

if mem is not contain "color" key, do nothing

if in "color" object contain "r" key, set r color of sketch to be mem["color"]["r"]

if in "color" object contain "g" key, set g color of sketch to be mem["color"]["g"]

if in "color" object contain "b" key, set b color of sketch to be mem["color"]["b"]

if in "color" object contain "a" key, set a color of sketch to be mem["color"]["a"]

example of param mem:

{

"color": {

```
 "r": 0,
 "g": 0,
 "b": 0,
 "a": 0
```

}

}

Parameters

mem	json, contain color of sketch

Definition at line 512 of file Sketch.cpp.

```
513 {
514     if(mem.contains("color"))
515     {
516         if(mem["color"].contains("r"))
517             color.r = mem["color"]["r"];
518         if(mem["color"].contains("g"))
519             color.g = mem["color"]["g"];
520         if(mem["color"].contains("b"))
521             color.b = mem["color"]["b"];
522         if(mem["color"].contains("a"))
523             color.a = mem["color"]["a"];
524         cache = color;
525     }
526 }
```

7.3.3.29 initFont()

```
void Sketch::initFont (
            const json & mem )  [protected], [inherited]
```

init font from json

if mem is not contain "font" key, do nothing

get font file and combine with GLOBAL::FontsFolder to get full path of font file

source font from that path and source the size of the font

if in "font" object contain "rect" key, get rect text

if in "font" object contain "color" key, get color text

if int "font" object contain "text", set default text of sketch to be mem["font"]["text"]

example of param mem:

{

"font": {

```
 "name": "font.ttf",
 "size": 0,
 "rect": {
      "x": 0,
      "y": 0,
      "w": 0,
      "h": 0
},
"color": {
      "r": 0,
      "g": 0,
      "b": 0,
      "a": 0
},
"text": "text"
```

}

}

Parameters

mem	json, contain font of sketch

Definition at line 584 of file Sketch.cpp.

```
585 {
586     if(!mem.contains("font")) return ;
587     if(mem["font"].contains("name") && mem["font"].contains("size"))
```

```
588     {
589         char* name = combineLink(GLOBAL::FontsFolder, mem["font"]["name"].get<std::string>().c_str());
590         if(font != nullptr)
591         {
592             TTF_CloseFont(font);
593             font = nullptr;
594         }
595         font = TTF_OpenFont(name, mem["font"]["size"]);
596     }
597     if(mem["font"].contains("rect"))
598     {
599         if(mem["font"]["rect"].contains("x"))
600             coor[1].x = mem["font"]["rect"]["x"];
601         if(mem["font"]["rect"].contains("y"))
602             coor[1].y = mem["font"]["rect"]["y"];
603         if(mem["font"]["rect"].contains("align X"))
604             textAlignX = mem["font"]["rect"]["align X"];
605         if(mem["font"]["rect"].contains("align Y"))
606             textAlignX = mem["font"]["rect"]["align Y"];
607     }
608     if(mem["font"].contains("color"))
609     {
610         if(mem["font"]["color"].contains("r"))
611         {
612             fontColor.r = mem["font"]["color"]["r"];
613         }
614         if(mem["font"]["color"].contains("g"))
615         {
616             fontColor.g = mem["font"]["color"]["g"];
617         }
618         if(mem["font"]["color"].contains("b"))
619         {
620             fontColor.b = mem["font"]["color"]["b"];
621         }
622         if(mem["font"]["color"].contains("a"))
623         {
624             fontColor.a = mem["font"]["color"]["a"];
625         }
626     }
627     if(mem["font"].contains("text"))
628     {
629         setText(mem["font"]["text"].get<std::string>());
630     }
631 }
```

7.3.3.30 isFocus()

```
bool Display::isFocus ( )
```

return true if user are iteractive on this display

Definition at line 18 of file Display.cpp.

```
19 {
20     return status;
21 }
```

7.3.3.31 isFreezed()

```
bool Display::isFreezed ( )
```

7.3.3.32 isLieInside()

```
bool Sketch::isLieInside (
            int x,
            int y )  [inherited]
```

determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

Parameters

x	int
y	int

Returns

bool

Definition at line 813 of file Sketch.cpp.

```
814 {
815     if(x < coor[0].x || coor[0].x + coor[0].w <= x) return false;
816     if(y < coor[0].y || coor[0].y + coor[0].y <= y) return false;
817     return true;
818 }
```

7.3.3.33 isLiesInside() [1/3]

```
bool Object::isLiesInside (
            int x,
            int y )  [inherited]
```

return true if a point lies inside this object

Definition at line 258 of file Object.cpp.

```
259 {
260     if(x < coor.x || coor.x + coor.w <= x)
261         return false;
262     if(y < coor.y || coor.y + coor.h <= y)
263         return false;
264     return true;
265 }
```

7.3.3.34 isLiesInside() [2/3]

```
bool Object::isLiesInside (
            int x,
            int y,
            int w,
            int h )  [inherited]
```

return true if a rectangle inside this object

Definition at line 269 of file Object.cpp.

```
270 {
271     if(x < coor.x || coor.x + coor.w <= x + w)
272         return false;
273     if(y < coor.y || coor.y + coor.h <= y + h)
274         return false;
275     return true;
276 }
```

7.3.3.35 isLiesInside() [3/3]

```
bool Object::isLiesInside (
            SDL_Rect rect )  [inherited]
```

return true if a rectangle inside this object

Definition at line 280 of file Object.cpp.

```
281 {
282     return isLiesInside(rect.x, rect.y, rect.w, rect.h);
283 }
```

7.3.3.36 isVisible()

```
bool Object::isVisible ( )  [inherited]
```

return true if this object is visible

Definition at line 250 of file Object.cpp.

```
251 {
252     return visable;
253 }
```

7.3.3.37 loadButton()

```
void Display::loadButton (
            Button *& but,
            const json & mem )
```

load button from json file

Parameters

but	button that will be loaded
mem	json file

Definition at line 161 of file Display.cpp.

```
162 {
163
164     if(but != nullptr)
165     {
166         delete but;
167         but = nullptr;
168     }
169
170     but = new Button;
171     but->setRenderer(ren);
172     but->init(
173             GLOBAL::AtrbButtons,
174             mem["name"].get<std::string>().c_str()
175             );
176     but->init(mem);
177     return ;
178 }
```

7.3.3.38 loadButtons()

```
void Display::loadButtons (
            const json & mem )
```

load buttons from json file

Parameters

mem	json file

Definition at line 75 of file Display.cpp.

```
76 {
77     DeleteButs();
78 
79     buts.resize(mem.size());
80 
81     for(int i = 0; i < (int) buts.size(); i++)
82     {
83         buts[i] = nullptr;
84         loadButton(buts[i], mem[i]);
85     }
86 }
```

7.3.3.39 mouseMove()

```
void Display::mouseMove (
            int x,
            int y )
```

handle mouse move event

Parameters

x	x coordinate of mouse
y	y coordinate of mouse if mouse is on button, it will change button's texture (if it has) and status

Definition at line 185 of file Display.cpp.

```
186 {
187     if(!isFocus()) return ;
188     if(!isVisible()) return ;
189 
190     for(int i = 0; i < (int) buts.size(); i++)
191         if(buts[i]->isChosen(x, y))
192             break;
193 }
```

7.3.3.40 mousePressedButton()

```
Button * Display::mousePressedButton (
            int x,
            int y )
```

get the button that is pressed

Parameters

x	x coordinate of mouse
y	y coordinate of mouse

Returns

button that is pressed or nullptr if no button is pressed

Definition at line 200 of file Display.cpp.

```
201 {
202     if(!isFocus()) return nullptr;
203     for(int i = 0; i < (int) buts.size(); i++)
204         if(buts[i]->isChosen(x, y))
205         {
206             return buts[i];
207         }
208     return nullptr;
209 }
```

7.3.3.41 moveTo()

```
void Display::moveTo (
              int x,
              int y,
              double time )
```

move the display to (x, y) coordinate in time (second) from the current position

Definition at line 283 of file Display.cpp.

```
284 {
285     int dx = x - getCoor().x;
286     int dy = y - getCoor().y;
287
288     if(diff(time, 0))
289     {
290         addX(dx);
291         addY(dy);
292
293         for(int i = 0; i < (int) buts.size(); i++)
294         {
295             buts[i]->addX(dx);
296             buts[i]->addY(dy);
297         }
298         return ;
299     }
300
301     double velo;
302
303     if(abs(dx) < abs(dy))
304         velo = dy / time;
305     else velo = dx / time;
306
307     int loop = std::min(80.0, abs(velo * time));
308
309     time = time / loop;
310
311     for(int i = 1; i <= loop; i++)
312     {
313         Uint32 startTime = SDL_GetTicks();
314
315         addX(-dx * (i - 1) / loop);
316         addX(dx * i / loop);
317         addY(-dy * (i - 1) / loop);
318         addY(dy * i / loop);
319
320         for(int j = 0; j < (int) buts.size(); j++)
321         {
322             buts[j]->addX(-dx * (i - 1) / loop);
```

```
323             buts[j]->addX(dx * i / loop);
324             buts[j]->addY(-dy * (i - 1) / loop);
325             buts[j]->addY(dy * i / loop);
326         }
327         Uint32 deltatime = SDL_GetTicks() - startTime;
328         SDL_Delay(time * 1000 - deltatime);
329     }
330 }
```

7.3.3.42 pickTexure()

```
void Object::pickTexure (
            int k )  [inherited]
```

choose top texture

Definition at line 231 of file Object.cpp.

```
232 {
233     if(k >= size()) return ;
234     top = k;
235 }
```

7.3.3.43 popChar()

```
void Sketch::popChar ( )  [inherited]
```

erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create

Definition at line 112 of file Sketch.cpp.

```
113 {
114     if(text.empty()) return ;
115     text.pop_back();
116     createTextTexture();
117 }
```

7.3.3.44 render() [1/2]

```
void Display::render ( )
```

render this display

Definition at line 98 of file Display.cpp.

```
99 {
100     if(!isVisible()) return ;
101     Object::render(0);
102
103     for(int i = 0; i < (int) buts.size(); i++)
104         buts[i]->render();
105 }
```

7.3.3.45 render() [2/2]

```
void Display::render (
            bool update )
```

render this display

Parameters

update	boolean if update is true, it will display to screen after render it

Definition at line 111 of file Display.cpp.

```
112 {
113     Object::render(update);
114     for(int i = 0; i < (int) buts.size(); i++)
115         buts[i]->render(update);
116 }
```

7.3.3.46 setBorder()

```
void Sketch::setBorder (
            int w,
            int r,
            int g,
            int b,
            int a )  [inherited]
```

set border

set border of sketch

Parameters

w	interger, width of border
r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255
a	interger, alpha value of border color, 0 - 255

Definition at line 355 of file Sketch.cpp.

```
356 {
357     borderWidth = w;
358     borderColor.r = r;
359     borderColor.g = g;
360     borderColor.b = b;
361     borderColor.a = a;
362 }
```

7.3.3.47 setBorderColor()

```
void Sketch::setBorderColor (
            int r,
            int g,
            int b )  [inherited]
```

set border color

Parameters

r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255 set border color

Definition at line 370 of file Sketch.cpp.

```
371 {
372     borderColor.r = r;
373     borderColor.g = g;
374     borderColor.b = b;
375 }
```

7.3.3.48 setColor() [1/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b )  [inherited]
```

set background color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set background color to be (r, g, b)

Definition at line 167 of file Sketch.cpp.

```
168 {
169     color.r = r;
170     color.g = g;
171     color.b = b;
172 }
```

7.3.3.49 setColor() [2/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b,
            int a )  [inherited]
```

set background color to be (r, g, b a)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255
a	interger, alpha value, 0 - 255 set background color to be (r, g, b, a)

Definition at line 182 of file Sketch.cpp.

```
183 {
184     color.r = r;
185     color.g = g;
186     color.b = b;
187     color.a = a;
188 }
```

7.3.3.50 setColor() [3/3]

```
void Sketch::setColor (
            SDL_Color c )  [inherited]
```

set background color to be c

Parameters

c	SDL_Color, background color

Definition at line 156 of file Sketch.cpp.

```
157 {
158     color = c;
159 }
```

7.3.3.51 setCoor() [1/2]

```
void Object::setCoor (
            int x,
            int y,
            int w,
            int h )  [inherited]
```

set coordinate of this object

Parameters

x	x coordinate
y	y coordinate
w	width
h	height

Definition at line 80 of file Object.cpp.

```
81 {
82     coor.x = x;
83     coor.y = y;
84     coor.w = w;
85     coor.h = h;
86 }
```

7.3.3.52 setCoor() [2/2]

```
void Object::setCoor (
            SDL_Rect key )  [inherited]
```

set coordinate of this object

Parameters

key	SDL_Rect

Definition at line 91 of file Object.cpp.

```
92 {
93     coor.x = key.x;
94     coor.y = key.y;
95     coor.w = key.w;
96     coor.h = key.h;
97 }
```

7.3.3.53 setH()

```
void Object::setH (
            int h )  [inherited]
```

set height of this object

Parameters

h	height

Definition at line 126 of file Object.cpp.

```
127 {
128     coor.h = h;
129 }
```

7.3.3.54 setInCenterX()

```
void Sketch::setInCenterX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the center of the background texture

Definition at line 269 of file Sketch.cpp.

```
270 {
271     int x = coor[0].x;
272     int w = coor[0].w;
273     coor[1].x = x + (w - coor[1].w) / 2;
274 }
```

7.3.3.55 setInCenterY()

```
void Sketch::setInCenterY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the center of the background texture

Definition at line 279 of file Sketch.cpp.

```
280 {
281     int y = coor[0].y;
282     int h = coor[0].h;
283     coor[1].y = y + (h - coor[1].h) / 2;
284 }
```

7.3.3.56 setOnLeftSideX()

```
void Sketch::setOnLeftSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the left side of the background texture

Definition at line 289 of file Sketch.cpp.

```
290 {
291     coor[1].x = coor[0].x;
292 }
```

7.3.3.57 setOnLeftSideY()

```
void Sketch::setOnLeftSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the top side of the background texture

Definition at line 307 of file Sketch.cpp.

```
308 {
309     coor[1].y = coor[0].y;
310 }
```

7.3.3.58 setOnRightSideX()

```
void Sketch::setOnRightSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the right side of the background texture

Definition at line 297 of file Sketch.cpp.

```
298 {
299     int x = coor[0].x;
300     int w = coor[0].w;
301     coor[1].x = x + w - coor[1].w;
302 }
```

7.3.3.59 setOnRightSideY()

```
void Sketch::setOnRightSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the bottom side of the background texture

Definition at line 315 of file Sketch.cpp.

```
316 {
317     int y = coor[0].y;
318     int h = coor[0].h;
319     coor[1].y = y + h - coor[1].h;
320 }
```

7.3.3.60 setRender()

```
void Sketch::setRender (
            SDL_Renderer *& r )  [inherited]
```

set render

Parameters

r	address of SDL_Renderer pointer set render of sketch

Definition at line 339 of file Sketch.cpp.

```
340 {
341     ren = r;
342 }
```

7.3.3.61 setRenderer()

```
void Display::setRenderer (
            SDL_Renderer *const & r )
```

set renderer for this display

Parameters

&r	renderer

Definition at line 91 of file Display.cpp.

```
92 {
93     ren = r;
94 }
```

7.3.3.62 setText()

```
void Sketch::setText (
            std::string s )  [inherited]
```

set text to be s

Parameters

s	string that will be set to text set text to be s and create new text texture

Definition at line 124 of file Sketch.cpp.

```
125 {
126     text = s;
127     createTextTexture();
128 }
```

7.3.3.63 setTextColor()

```
void Sketch::setTextColor (
            int r,
            int g,
            int b )  [inherited]
```

set text color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set text color to be (r, g, b) and create new text texture

Definition at line 136 of file Sketch.cpp.

```
137 {
138     fontColor.r = r;
139     fontColor.g = g;
140     fontColor.b = b;
141     createTextTexture();
142 }
```

7.3.3.64 setTextures()

```
void Object::setTextures (
            const json & mem )  [inherited]
```

load textures from json file

Definition at line 154 of file Object.cpp.

```
155 {
156     clearTextures();
157
158     tes.resize(mem["textures"].size());
159
160     char* FolderName = new char [256];
161     strcpy(FolderName, mem["name"].get<std::string>().c_str());
162
163     for(int i = 0 ; i < size(); i++)
164     {
165         const char* fullname = combineName(
166             mem["textures"][i]["name"].get<std::string>().c_str(),
```

```
167             mem["textures"][i]["type"].get<std::string>().c_str()
168         );
169         const char* name = combineLink(
170             FolderName,
171             fullname
172         );
173         const char* link = combineLink(
174             GLOBAL::GraphicsFolder,
175             name
176         );
177 
178         SDL_Surface* surf;
179 
180         std::string type = mem["textures"][i]["type"].get<std::string>();
181         if(type == "bmp")
182             surf = SDL_LoadBMP(link);
183         else if(type == "png" || type == "jpg")
184             surf = IMG_Load(link);
185 
186         tes[i] = SDL_CreateTextureFromSurface(ren, surf);
187 
188         delete [] link;
189         delete [] name;
190         delete []fullname;
191         SDL_FreeSurface(surf);
192     }
193     delete [] FolderName;
194 }
```

7.3.3.65 setW()

```
void Object::setW (
              int w )  [inherited]
```

set width of this object

Parameters

w	width

Definition at line 118 of file Object.cpp.

```
119 {
120     coor.w = w;
121 }
```

7.3.3.66 setX()

```
void Object::setX (
              int x )  [inherited]
```

set x coordinate of this object

Parameters

x	x coordinate

Definition at line 102 of file Object.cpp.

```
103 {
```

```
104     coor.x = x;
105 }
```

7.3.3.67 setY()

```
void Object::setY (
            int y )  [inherited]
```

set y coordinate of this object

Parameters

y	y coordinate

Definition at line 110 of file Object.cpp.

```
111 {
112     coor.y = y;
113 }
```

7.3.3.68 show()

```
void Object::show ( )  [inherited]
```

show this object

Definition at line 140 of file Object.cpp.

```
141 {
142     visable = true;
143 }
```

7.3.3.69 showButton()

```
void Display::showButton (
            int k )
```

set button visible to true

Parameters

k	index of button

Definition at line 142 of file Display.cpp.

```
143 {
144     if(k >= (int) buts.size()) return ;
145     buts[k]->show();
146 }
```

7.3.3.70 size()

```
int Object::size ( )  [inherited]
```

return number of textures

Definition at line 208 of file Object.cpp.

```
209 {
210     return tes.size();
211 }
```

7.3.3.71 trigger()

```
void Display::trigger (
            int x,
            int y )
```

Run animation when mouse is in trigger area of screen.

Parameters

x	x coordinate of mouse
y	y coordinate of mouse

Definition at line 336 of file Display.cpp.

```
337 {
338     if(!isLiesInside(x, y)) return ;
339
340     if(!isVisible())
341     {
342         int dy = 100;
343         addY(dy);
344         for(int i = 0; i < buts.size(); i++)
345             buts[i]->addY(dy);
346         show();
347         moveTo(260, 440, 0.4);
348     }
349     if(!isFocus() && isVisible())
350     {
351         moveTo(260, 440, 0.4);
352         hide();
353     }
354 }
```

7.3.3.72 unHighlight()

```
void Sketch::unHighlight ( )  [inherited]
```

unhightlight the sketch this function will change color of background to normal color

Definition at line 884 of file Sketch.cpp.

```
885 {
886     color = cache;
887     FillWithColor();
888 }
```

The documentation for this class was generated from the following files:

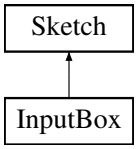
- include/Display.hpp
- src/Display.cpp

7.4 InputBox Class Reference

class that create an input box and render it to the screen Popup a box that can typing in.

`#include <InputBox.hpp>`

Inheritance diagram for InputBox:



Public Member Functions

- InputBox ()
  constructor of InputBox
- ~InputBox ()
- void setRender (SDL_Renderer *&r)
  set renderer for this InputBox
- void init (const json &mem)
  load this object from json file
- void typing (char ch)
  add a character to chosen input area
- void setInput (std::string s)
  set text for chosen input area
- void pop ()
  pop a character from chosen input area
- void render ()
  render this InputBox
- void mouseMove (int x, int y)
  handle mouse move event
- void mousePress (int x, int y)
  choose input area
- Button * getButtonPressedByMouse (int x, int y)
  return button pressed by mouse
- void nextFocus ()
  choose input area
- void setFocus (int k)
  choose input area
- std::string getText (int k)
  return text of chosen input area
- bool isVisible ()
  get visible this function will return visible of sketch
- void show ()
  show the sketch this function will set visible to true, that will enable the sketch to be rendered
- void hide ()
  hide the sketch this function will set visible to false, that will disable the sketch to be rendered
- void addChar (char ch)

typing text

- void popChar ()

  erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create

- void setText (std::string s)

  set text to be s

- void setTextColor (int r, int g, int b)

  set text color to be (r, g, b)

- const std::string & getText ()

  return text

- void setColor (SDL_Color c)

  set background color to be c

- void setColor (int r, int g, int b)

  set background color to be (r, g, b)

- void setColor (int r, int g, int b, int a)

  set background color to be (r, g, b a)

- void setCoor (int x, int y, int w, int h)

  set coordinate of sketch

- void setX (int x)

  set coordinate of sketch

- void setY (int y)

  set coordinate of sketch

- void setW (int w)

  set coordinate of sketch

- void setH (int h)

  set coordinate of sketch

- void addX (int x)

  set coordinate of sketch

- void addY (int y)

  set coordinate of sketch

- void setInCenterX ()

  align text texture align x coordinate of text texture to be in the center of the background texture

- void setOnLeftSideX ()

  align text texture align x coordinate of text texture to be in the left side of the background texture

- void setOnRightSideX ()

  align text texture align x coordinate of text texture to be in the right side of the background texture

- void setInCenterY ()

  align text texture align y coordinate of text texture to be in the center of the background texture

- void setOnLeftSideY ()

  align text texture align y coordinate of text texture to be in the top side of the background texture

- void setOnRightSideY ()

  align text texture align y coordinate of text texture to be in the bottom side of the background texture

- void align ()

  align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

- SDL_Rect getCoor ()

  get coordinate this function will return coordinate of background of sketch

- void setBorder (int w, int r, int g, int b, int a)

  set border

- void setBorderColor (int r, int g, int b)

  set border color

- void FillWithColor ()

  fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

- void FillWithColor (SDL_Color c)

  fill background color with color C

- void highlight ()

  hightlight the sketch this function will change color of background to invert color

- void unHighlight ()

  unhightlight the sketch this function will change color of background to normal color

- bool isLieInside (int x, int y)

  determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

- void moveTo (int x, int y, double time)

  animation of sketch to move the sketch to point (x, y) in time (second) this function will move the sketch to point (x, y) in time (second)

Protected Member Functions

- void clearTexture (int k)

  clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault

- void initRect (const json &mem)

  set cooridnate of sketch from json

- void initColor (const json &mem)

  init color from json

- void initFont (const json &mem)

  init font from json

- void initBorder (const json &mem)

  init border from json

- void createTextTexture ()

  create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

7.4.1 Detailed Description

class that create an input box and render it to the screen Popup a box that can typing in.

Definition at line 14 of file InputBox.hpp.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 InputBox()

```
InputBox::InputBox ( )
```

constructor of InputBox

Definition at line 5 of file InputBox.cpp.

```
6 {
7     ren = nullptr;
8     boxTitle = nullptr;
9     input.clear();
10     texts.clear();
11     buts.clear();
12     focusOn = -1;
13 }
```

7.4.2.2 ~InputBox()

```
InputBox::~InputBox ( )
```

Definition at line 15 of file InputBox.cpp.

```
16 {
17     ren = nullptr;
18     if(boxTitle != nullptr)
19     {
20         delete boxTitle;
21         boxTitle = nullptr;
22     }
23     input.clear();
24     texts.clear();
25     buts.clear();
26 }
```

7.4.3 Member Function Documentation

7.4.3.1 addChar()

```
void Sketch::addChar (
            char ch )  [inherited]
```

typing text

Parameters

ch	character that will be add to the end of the text add a character to the end of the text after that new text texture will be created

Definition at line 101 of file Sketch.cpp.

```
102 {
103     text = text + ch;
104     createTextTexture();
105 }
```

7.4.3.2 addX()

```
void Sketch::addX (
            int x )  [inherited]
```

set coordinate of sketch

Parameters

x,interger,change	of x coordinate of the top left corner of the sketch add x to x coordinate of sketch

Definition at line 218 of file Sketch.cpp.

```
219 {
220     coor[0].x += x;
221     align();
222 }
```

7.4.3.3 addY()

```
void Sketch::addY (
            int y )  [inherited]
```

set coordinate of sketch

Parameters

y	interger, change of y coordinate of the top left corner of the sketch add y to y coordinate of sketch

Definition at line 229 of file Sketch.cpp.

```
230 {
231     coor[0].y += y;
232     align();
233 }
```

7.4.3.4 align()

```
void Sketch::align ( )  [inherited]
```

align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

Definition at line 761 of file Sketch.cpp.

```
762 {
763
764     if(textAlignX == 1) setOnLeftSideX();
765     if(textAlignX == 2) setInCenterX();
766     if(textAlignX == 3) setOnRightSideX();
767
768     if(textAlignY == 1) setOnLeftSideY();
769     if(textAlignY == 2) setInCenterY();
770     if(textAlignY == 3) setOnRightSideY();
771 }
```

7.4.3.5 clearTexture()

```
void Sketch::clearTexture (
            int k )  [protected], [inherited]
```

clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault

Parameters

k	integer, index of textures, 0 will be background, 1 will be text if tes[k] is nullptr, do nothing call SDL_DestroyTexture and after that set tes[k] to be nullptr

Definition at line 37 of file Sketch.cpp.

```
38 {
39     if(tes[k] == nullptr) return ;
40     SDL_DestroyTexture(tes[k]);
41     tes[k] = nullptr;
42 }
```

7.4.3.6 createTextTexture()

```
void Sketch::createTextTexture ( )  [protected], [inherited]
```

create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

Definition at line 63 of file Sketch.cpp.

```
64 {
65     clearTexture(1);
66     if(text.empty()) return ;
67 
68     SDL_Surface* surface = TTF_RenderText_Solid(font, text.c_str(), fontColor);
69 
70     tes[1] = SDL_CreateTextureFromSurface(ren, surface);
71 
72     coor[1].w = surface->w;
73     coor[1].h = surface->h;
74 
75     crop = coor[1];
76     crop.x = 0;
77     crop.y = 0;
78 
79     if(coor[1].w > coor[0].w || coor[1].h > coor[0].h)
80     {
81         crop = SDL_Rect({
82                 std::max(0, coor[1].w - coor[0].w),
83                 std::max(0, coor[1].h - coor[0].h),
84                 coor[0].w,
85                 coor[0].h
86             });
87         coor[1].w = coor[0].w;
88         coor[1].h = coor[0].h;
89     }
90 
91     align();
92 
93     SDL_FreeSurface(surface);
94 }
```

7.4.3.7 FillWithColor() [1/2]

```
void Sketch::FillWithColor ( )  [inherited]
```

fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

Definition at line 394 of file Sketch.cpp.

```
395 {
396     int w = coor[0].w;
397     int h = coor[0].h;
398     clearTexture(0);
399 
400     SDL_Surface* surf = SDL_CreateRGBSurfaceWithFormat(0, w, h, 32, SDL_PIXELFORMAT_RGBA32);
401     SDL_SetSurfaceBlendMode(surf, SDL_BLENDMODE_BLEND);
402 
403     SDL_FillRect(surf, nullptr, SDL_MapRGBA(surf->format, color.r, color.g, color.b, color.a));
404 
405     SDL_Rect borderRect;
406 
407     Uint32 c = SDL_MapRGBA(surf->format, borderColor.r, borderColor.g, borderColor.b, borderColor.a);
408     borderRect = SDL_Rect({0, 0, borderWidth, h});
409     SDL_FillRect(surf, &borderRect, c);
410 
411     borderRect = SDL_Rect({0, 0, w, borderWidth});
412     SDL_FillRect(surf, &borderRect, c);
413 
414     borderRect = SDL_Rect({0, h - borderWidth, w, borderWidth});
415     SDL_FillRect(surf, &borderRect, c);
416 
417     borderRect = SDL_Rect({w - borderWidth, 0, borderWidth, h});
418     SDL_FillRect(surf, &borderRect, c);
419 
420     tes[0] = SDL_CreateTextureFromSurface(ren, surf);
421 
422     SDL_FreeSurface(surf);
423 
424 }
```

7.4.3.8 FillWithColor() [2/2]

```
void Sketch::FillWithColor (
            SDL_Color c )  [inherited]
```

fill background color with color C

Parameters

c	SDL_Color, color to fill fill background color with color C

Definition at line 381 of file Sketch.cpp.

```
382 {
383     SDL_Color temp = color;
384     color = c;
385     FillWithColor();
386     color = temp;
387 }
```

7.4.3.9 getButtonPressedByMouse()

```
Button * InputBox::getButtonPressedByMouse (
```

```
            int x,
            int y )
```

return button pressed by mouse

Definition at line 180 of file InputBox.cpp.

```
181 {
182     if(!isVisible()) return nullptr;
183 
184     for(int i = 0; i < (int) buts.size(); i++)
185         if(buts[i]->isChosen(x, y))
186             return buts[i];
187     return nullptr;
188 }
```

7.4.3.10 getCoor()

```
SDL_Rect Sketch::getCoor ( )  [inherited]
```

get coordinate this function will return coordinate of background of sketch

Returns

SDL_Rect

Definition at line 777 of file Sketch.cpp.

```
778 {
779     return coor[0];
780 }
```

7.4.3.11 getText() [1/2]

```
const std::string & Sketch::getText ( )  [inherited]
```

return text

Definition at line 148 of file Sketch.cpp.

```
149 {
150     return text;
151 }
```

7.4.3.12 getText() [2/2]

```
std::string InputBox::getText (
            int k )
```

return text of chosen input area

Definition at line 193 of file InputBox.cpp.

```
194 {
195     return input[k]->getText();
196 }
```

7.4.3.13 hide()

```
void Sketch::hide ( )  [inherited]
```

hide the sketch this function will set visible to false, that will disable the sketch to be rendered

Definition at line 802 of file Sketch.cpp.

```
803 {
804     visible = false;
805 }
```

7.4.3.14 highlight()

```
void Sketch::highlight ( )  [inherited]
```

hightlight the sketch this function will change color of background to invert color

Definition at line 867 of file Sketch.cpp.

```
868 {
869     color.r = 255 - color.r;
870     color.g = 255 - color.g;
871     color.b = 255 - color.g;
872     if(color.r > 20 && color.g > 20 && color.b > 20)
873     {
874         color.r -= color.r * 0.3;
875         color.g -= color.g * 0.3;
876         color.b -= color.b * 0.3;
877     }
878     FillWithColor();
879 }
```

7.4.3.15 init()

```
void InputBox::init (
            const json & mem )
```

load this object from json file

Definition at line 37 of file InputBox.cpp.

```
38 {
39     focusOn = -1;
40     Sketch::setRender(ren);
41     Sketch::init(mem);
42     if(mem.contains("title"))
43     {
44         boxTitle = new Sketch;
45         boxTitle->setRender(ren);
46         boxTitle->init(mem["title"]);
47     }
48     if(mem.contains("input"))
49     {
50         input.clear();
51         input.resize(mem["input"].size());
52         for(int i = 0; i < (int) input.size(); i++)
53         {
54             input[i] = new Sketch;
55             input[i]->setRender(ren);
56             input[i]->init(mem["input"][i]);
57         }
58     }
59     if(mem.contains("texts"))
60     {
61         texts.clear();
62         texts.resize(mem["texts"].size());
```

```
63          for(int i = 0; i < (int) texts.size(); i++)
64          {
65              texts[i] = new Sketch;
66              texts[i]->setRender(ren);
67              texts[i]->init(mem["texts"][i]);
68          }
69      }
70      if(mem.contains("buttons"))
71      {
72          buts.clear();
73          buts.resize(mem["buttons"].size());
74          for(int i = 0; i < (int) buts.size(); i++)
75          {
76              buts[i] = new Button;
77              buts[i]->setRenderer(ren);
78              buts[i]->init(
79                      GLOBAL::AtrbButtons,
80                      mem["buttons"][i]["name"].get<std::string>().c_str()
81                      );
82              buts[i]->init(mem["buttons"][i]);
83          }
84      }
85  }
```

7.4.3.16 initBorder()

```
void Sketch::initBorder (
              const json & mem )  [protected], [inherited]
```

init border from json

if mem is not contain "border" key, do nothing

if in "border" object contain "width" key, set width of border to be mem["border"]["width"]

if in "border" object contain "color" key, set color of border to be mem["border"]["color"]

example of param mem:

{

"border": {

```
"width": 0,

"color": {

  "r": 0,

  "g": 0,

  "b": 0,

  "a": 0

}
```

}

}

Parameters

mem	json, contain border of sketch

Definition at line 667 of file Sketch.cpp.

```
668 {
669     if(!mem.contains("border")) return ;
670     if(mem["border"].contains("width"))
671         borderWidth = mem["border"]["width"];
672 
673     if(mem["border"].contains("color"))
674     {
675         if(mem["border"]["color"].contains("r"))
676         {
677             borderColor.r = mem["border"]["color"]["r"];
678         }
679         if(mem["border"]["color"].contains("g"))
680         {
681             borderColor.g = mem["border"]["color"]["g"];
682         }
683         if(mem["border"]["color"].contains("b"))
684         {
685             borderColor.b = mem["border"]["color"]["b"];
686         }
687         if(mem["border"]["color"].contains("a"))
688         {
689             borderColor.a = mem["border"]["color"]["a"];
690         }
691     }
692 }
```

7.4.3.17 initColor()

```
void Sketch::initColor (
            const json & mem )  [protected], [inherited]
```

init color from json

if mem is not contain "color" key, do nothing

if in "color" object contain "r" key, set r color of sketch to be mem["color"]["r"]

if in "color" object contain "g" key, set g color of sketch to be mem["color"]["g"]

if in "color" object contain "b" key, set b color of sketch to be mem["color"]["b"]

if in "color" object contain "a" key, set a color of sketch to be mem["color"]["a"]

example of param mem:

{

"color": {

```
 "r": 0,
 "g": 0,
 "b": 0,
 "a": 0
```

}

}

Parameters

mem	json, contain color of sketch

Definition at line 512 of file Sketch.cpp.

```
513 {
514     if(mem.contains("color"))
515     {
516         if(mem["color"].contains("r"))
517             color.r = mem["color"]["r"];
518         if(mem["color"].contains("g"))
519             color.g = mem["color"]["g"];
520         if(mem["color"].contains("b"))
521             color.b = mem["color"]["b"];
522         if(mem["color"].contains("a"))
523             color.a = mem["color"]["a"];
524         cache = color;
525     }
526 }
```

7.4.3.18 initFont()

```
void Sketch::initFont (
            const json & mem )  [protected], [inherited]
```

init font from json

if mem is not contain "font" key, do nothing

get font file and combine with GLOBAL::FontsFolder to get full path of font file

source font from that path and source the size of the font

if in "font" object contain "rect" key, get rect text

if in "font" object contain "color" key, get color text

if int "font" object contain "text", set default text of sketch to be mem["font"]["text"]

example of param mem:

{

"font": {

```
 "name": "font.ttf",
 "size": 0,
 "rect": {
      "x": 0,
      "y": 0,
      "w": 0,
      "h": 0
},
"color": {
```

```
      "r": 0,

      "g": 0,

      "b": 0,

      "a": 0

},

"text": "text"


}

}
```

Parameters

mem	json, contain font of sketch

Definition at line 584 of file Sketch.cpp.

```
585 {
586     if(!mem.contains("font")) return ;
587     if(mem["font"].contains("name") && mem["font"].contains("size"))
588     {
589         char* name = combineLink(GLOBAL::FontsFolder, mem["font"]["name"].get<std::string>().c_str());
590         if(font != nullptr)
591         {
592             TTF_CloseFont(font);
593             font = nullptr;
594         }
595         font = TTF_OpenFont(name, mem["font"]["size"]);
596     }
597     if(mem["font"].contains("rect"))
598     {
599         if(mem["font"]["rect"].contains("x"))
600             coor[1].x = mem["font"]["rect"]["x"];
601         if(mem["font"]["rect"].contains("y"))
602             coor[1].y = mem["font"]["rect"]["y"];
603         if(mem["font"]["rect"].contains("align X"))
604             textAlignX = mem["font"]["rect"]["align X"];
605         if(mem["font"]["rect"].contains("align Y"))
606             textAlignX = mem["font"]["rect"]["align Y"];
607     }
608     if(mem["font"].contains("color"))
609     {
610         if(mem["font"]["color"].contains("r"))
611         {
612             fontColor.r = mem["font"]["color"]["r"];
613         }
614         if(mem["font"]["color"].contains("g"))
615         {
616             fontColor.g = mem["font"]["color"]["g"];
617         }
618         if(mem["font"]["color"].contains("b"))
619         {
620             fontColor.b = mem["font"]["color"]["b"];
621         }
622         if(mem["font"]["color"].contains("a"))
623         {
624             fontColor.a = mem["font"]["color"]["a"];
625         }
626     }
627     if(mem["font"].contains("text"))
628     {
629         setText(mem["font"]["text"].get<std::string>());
630     }
631 }
```

7.4.3.19 initRect()

```
void Sketch::initRect (
```

```
const json & mem )  [protected], [inherited]
```

set cooridnate of sketch from json

if mem is not contain "rect" key, do nothing

if in "rect" object contain "x" key, set x coordinate of sketch to be mem["rect"]["x"]

if in "rect" object contain "y" key, set y coordinate of sketch to be mem["rect"]["y"]

if in "rect" object contain "w" key, set w coordinate of sketch to be mem["rect"]["w"]

if in "rect" object contain "h" key, set h coordinate of sketch to be mem["rect"]["h"]

example of param mem:

{

```
"rect": {
     "x": 0,
     "y": 0,
     "w": 0,
     "h": 0
}
```

Parameters

mem	json, contain coordinate of sketch

Definition at line 457 of file Sketch.cpp.

```
458 {
459     if(mem.contains("rect"))
460     {
461         if(mem["rect"].contains("x"))
462         {
463             coor[0].x = mem["rect"]["x"];
464         }
465         if(mem["rect"].contains("y"))
466         {
467             coor[0].y = mem["rect"]["y"];
468         }
469         if(mem["rect"].contains("w"))
470         {
471             coor[0].w = mem["rect"]["w"];
472         }
473         if(mem["rect"].contains("h"))
474         {
475             coor[0].h = mem["rect"]["h"];
476         }
477     }
478 }
```

7.4.3.20 isLieInside()

```
bool Sketch::isLieInside (
             int x,
             int y )  [inherited]
```

determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

Parameters

x	int
y	int

Returns

bool

Definition at line 813 of file Sketch.cpp.

```
814 {
815     if(x < coor[0].x || coor[0].x + coor[0].w <= x) return false;
816     if(y < coor[0].y || coor[0].y + coor[0].y <= y) return false;
817     return true;
818 }
```

7.4.3.21 isVisible()

```
bool Sketch::isVisible ( )  [inherited]
```

get visible this function will return visible of sketch

Returns

bool

Definition at line 786 of file Sketch.cpp.

```
787 {
788     return visible;
789 }
```

7.4.3.22 mouseMove()

```
void InputBox::mouseMove (
            int x,
            int y )
```

handle mouse move event

Definition at line 144 of file InputBox.cpp.

```
145 {
146     if(!isVisible()) return ;
147 
148     for(int i = 0; i < (int) buts.size(); i++)
149         if(buts[i]->isChosen(x, y)) break;
150 
151 }
```

7.4.3.23 mousePress()

```
void InputBox::mousePress (
            int x,
            int y )
```

choose input area

Definition at line 155 of file InputBox.cpp.

```
156 {
157     if(!isVisible()) return ;
158     for(int i = 0; i < (int) input.size(); i++)
159         if(input[i]->isLieInside(x, y))
160             setFocus(i);
161     return ;
162 }
```

7.4.3.24 moveTo()

```
void Sketch::moveTo (
            int x,
            int y,
            double time )  [inherited]
```

animation of sketch to move the sketch to point (x, y) in time (second) this function will move the sketch to point (x, y) in time (second)

Parameters

x	int
y	int
time	double

Definition at line 826 of file Sketch.cpp.

```
827 {
828     int dx = x - getCoor().x;
829     int dy = y - getCoor().y;
830
831     if(diff(time, 0))
832     {
833         setX(x);
834         setY(y);
835         return ;
836     }
837
838     double velo;
839
840     if(abs(dx) < abs(dy))
841         velo = dy / time;
842     else velo = dx / time;
843
844     int loop = std::min(80.0, abs(velo * time));
845
846     time = time / loop;
847
848     for(int i = 0; i <= loop; i++)
849     {
850         Uint32 startTime = SDL_GetTicks();
851
852         addX(-dx * (i - 1) / loop);
853         addX(dx * i / loop);
854         addY(-dy * (i - 1) / loop);
855         addY(dy * i / loop);
856         render();
857         Uint32 deltaTime = SDL_GetTicks() - startTime;
```

```
858         startTime = SDL_GetTicks();
859         if(deltaTime <= time * 1000)
860             SDL_Delay(time * 1000 - deltaTime);
861     }
862 }
```

7.4.3.25 nextFocus()

```
void InputBox::nextFocus ( )
```

choose input area

Definition at line 166 of file InputBox.cpp.

```
167 {
168     if(!isVisible()) return ;
169     if(focusOn == -1)
170     {
171         if(!input.empty()) focusOn = 0;
172         return ;
173     }
174     if(input.empty()) return ;
175     focusOn = (focusOn + 1) % input.size();
176 }
```

7.4.3.26 pop()

```
void InputBox::pop ( )
```

pop a character from chosen input area

Definition at line 134 of file InputBox.cpp.

```
135 {
136     if(!isVisible()) return ;
137     if(focusOn == -1) return ;
138     input[focusOn]->popChar();
139 }
```

7.4.3.27 popChar()

```
void Sketch::popChar ( )  [inherited]
```

erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create

Definition at line 112 of file Sketch.cpp.

```
113 {
114     if(text.empty()) return ;
115     text.pop_back();
116     createTextTexture();
117 }
```

7.4.3.28 render()

```
void InputBox::render ( )
```

render this InputBox

Definition at line 89 of file InputBox.cpp.

```
90 {
91     if(!isVisible()) return ;
92
93     Sketch::render();
94
95     if(boxTitle != nullptr)
96     {
97         boxTitle->render();
98     }
99     for(int i = 0; i < (int) input.size(); i++)
100         input[i]->render();
101     for(int i = 0; i < (int) texts.size(); i++)
102         texts[i]->render();
103     for(int i = 0; i < (int) buts.size(); i++)
104         buts[i]->render();
105 }
```

7.4.3.29 setBorder()

```
void Sketch::setBorder (
            int w,
            int r,
            int g,
            int b,
            int a )  [inherited]
```

set border

set border of sketch

Parameters

w	interger, width of border
r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255
a	interger, alpha value of border color, 0 - 255

Definition at line 355 of file Sketch.cpp.

```
356 {
357     borderWidth = w;
358     borderColor.r = r;
359     borderColor.g = g;
360     borderColor.b = b;
361     borderColor.a = a;
362 }
```

7.4.3.30 setBorderColor()

```
void Sketch::setBorderColor (
            int r,
```

```
            int g,
            int b )  [inherited]
```

set border color

Parameters

r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255 set border color

Definition at line 370 of file Sketch.cpp.

```
371 {
372     borderColor.r = r;
373     borderColor.g = g;
374     borderColor.b = b;
375 }
```

7.4.3.31 setColor() [1/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b )  [inherited]
```

set background color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set background color to be (r, g, b)

Definition at line 167 of file Sketch.cpp.

```
168 {
169     color.r = r;
170     color.g = g;
171     color.b = b;
172 }
```

7.4.3.32 setColor() [2/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b,
            int a )  [inherited]
```

set background color to be (r, g, b a)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255
a	interger, alpha value, 0 - 255 set background color to be (r, g, b, a)

Definition at line 182 of file Sketch.cpp.

```
183 {
184     color.r = r;
185     color.g = g;
186     color.b = b;
187     color.a = a;
188 }
```

7.4.3.33 setColor() [3/3]

```
void Sketch::setColor (
            SDL_Color c )  [inherited]
```

set background color to be c

Parameters

c	SDL_Color, background color

Definition at line 156 of file Sketch.cpp.

```
157 {
158     color = c;
159 }
```

7.4.3.34 setCoor()

```
void Sketch::setCoor (
            int x,
            int y,
            int w,
            int h )  [inherited]
```

set coordinate of sketch

Parameters

x	interger, x coordinate of the top left corner of the sketch
y	interger, y coordinate of the top left corner of the sketch
w	interger, width of the sketch
h	interger, height of the sketch set coordinate of sketch

Definition at line 198 of file Sketch.cpp.

```
199 {
200     coor[0] = SDL_Rect({x, y, w, h});
201     align();
202 }
```

7.4.3.35 setFocus()

```
void InputBox::setFocus (
            int k )
```

choose input area

Definition at line 109 of file InputBox.cpp.

```
110 {
111     if(!isVisible()) return ;
112     if(k >= input.size()) return ;
113     focusOn = k;
114 }
```

7.4.3.36 setH()

```
void Sketch::setH (
            int h )  [inherited]
```

set coordinate of sketch

Parameters

h	interger, height of the sketch set height of sketch

Definition at line 260 of file Sketch.cpp.

```
261 {
262     coor[0].h = h;
263     align();
264 }
```

7.4.3.37 setInCenterX()

```
void Sketch::setInCenterX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the center of the background texture

Definition at line 269 of file Sketch.cpp.

```
270 {
271     int x = coor[0].x;
272     int w = coor[0].w;
273     coor[1].x = x + (w - coor[1].w) / 2;
274 }
```

7.4.3.38 setInCenterY()

```
void Sketch::setInCenterY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the center of the background texture

Definition at line 279 of file Sketch.cpp.

```
280 {
281     int y = coor[0].y;
282     int h = coor[0].h;
283     coor[1].y = y + (h - coor[1].h) / 2;
284 }
```

7.4.3.39 setInput()

```
void InputBox::setInput (
            std::string s )
```

set text for chosen input area

Definition at line 126 of file InputBox.cpp.

```
127 {
128     if(focusOn == -1) return ;
129     input[focusOn]->setText(s);
130 }
```

7.4.3.40 setOnLeftSideX()

```
void Sketch::setOnLeftSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the left side of the background texture

Definition at line 289 of file Sketch.cpp.

```
290 {
291     coor[1].x = coor[0].x;
292 }
```

7.4.3.41 setOnLeftSideY()

```
void Sketch::setOnLeftSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the top side of the background texture

Definition at line 307 of file Sketch.cpp.

```
308 {
309     coor[1].y = coor[0].y;
310 }
```

7.4.3.42 setOnRightSideX()

```
void Sketch::setOnRightSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the right side of the background texture

Definition at line 297 of file Sketch.cpp.

```
298 {
299     int x = coor[0].x;
300     int w = coor[0].w;
301     coor[1].x = x + w - coor[1].w;
302 }
```

7.4.3.43 setOnRightSideY()

```
void Sketch::setOnRightSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the bottom side of the background texture

Definition at line 315 of file Sketch.cpp.

```
316 {
317     int y = coor[0].y;
318     int h = coor[0].h;
319     coor[1].y = y + h - coor[1].h;
320 }
```

7.4.3.44 setRender()

```
void InputBox::setRender (
            SDL_Renderer *& r )
```

set renderer for this InputBox

Definition at line 30 of file InputBox.cpp.

```
31 {
32     ren = r;
33 }
```

7.4.3.45 setText()

```
void Sketch::setText (
            std::string s )  [inherited]
```

set text to be s

Parameters

s	string that will be set to text set text to be s and create new text texture

Definition at line 124 of file Sketch.cpp.

```
125 {
126     text = s;
127     createTextTexture();
128 }
```

7.4.3.46 setTextColor()

```
void Sketch::setTextColor (
            int r,
            int g,
            int b )  [inherited]
```

set text color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set text color to be (r, g, b) and create new text texture

Definition at line 136 of file Sketch.cpp.

```
137 {
138     fontColor.r = r;
139     fontColor.g = g;
140     fontColor.b = b;
141     createTextTexture();
142 }
```

7.4.3.47 setW()

```
void Sketch::setW (
            int w )  [inherited]
```

set coordinate of sketch

Parameters

w	interger, width of the sketch set width of sketch

Definition at line 250 of file Sketch.cpp.

```
251 {
252     coor[0].w = w;
253     align();
254 }
```

7.4.3.48 setX()

```
void Sketch::setX (
            int x )  [inherited]
```

set coordinate of sketch

Parameters

x	interger, x coordinate of the top left corner of the sketch set x coordinate of sketch

Definition at line 208 of file Sketch.cpp.

```
209 {
210     coor[0].x = x;
211     align();
212 }
```

7.4.3.49 setY()

```
void Sketch::setY (
            int y )  [inherited]
```

set coordinate of sketch

Parameters

y	interger, y coordinate of the top left corner of the sketch set y coordinate of sketch

Definition at line 240 of file Sketch.cpp.

```
241 {
242     coor[0].y = y;
243     align();
244 }
```

7.4.3.50 show()

```
void Sketch::show ( )  [inherited]
```

show the sketch this function will set visible to true, that will enable the sketch to be rendered

Definition at line 794 of file Sketch.cpp.

```
795 {
796     visible = true;
797 }
```

7.4.3.51 typing()

```
void InputBox::typing (
            char ch )
```

add a character to chosen input area

Definition at line 118 of file InputBox.cpp.

```
119 {
120     if(focusOn == -1) return ;
121     input[focusOn]->addChar(ch);
122 }
```

7.4.3.52 unHighlight()

```
void Sketch::unHighlight ( )  [inherited]
```

unhightlight the sketch this function will change color of background to normal color

Definition at line 884 of file Sketch.cpp.

```
885 {
886     color = cache;
887     FillWithColor();
888 }
```

The documentation for this class was generated from the following files:

- include/InputBox.hpp
- src/InputBox.cpp

7.5 MyWindow Class Reference

class that handle screen box, input box, data_structures box Finite state machine

`#include <DuckWin.hpp>`

Public Member Functions

- MyWindow ()
  contructor of MyWindow
- void loadScreen (Display *&screen, const json &mem)
- void init ()
  init the window default size is 960x540
- void action ()
  get user input
- void process ()
  Handle user input.
- void speak ()
  sound function
- void render ()
  render function
- bool isOpen ()
  Return true if MyWindow is open, false otherwise.
- bool isClose ()
  Return true if MyWindow is close, false otherwise.
- bool isHanging ()
- void changeScreens (const char *const &name)
  Delete all screens and add new screens which information is in json file have directory is "GLOBAL::Atrb↩ Screens/name".
- Display *& top ()
  Return the screen that is focus on.
- void mouseMove (int x, int y)
  when user move mouse, this function will be called
- void mousePress (int x, int y)
  When user press mouse, this function will be called to handle it.
- void typing (SDL_Keysym key)
  add a character to input box
- void run ()
  Start MyWindow render and sound, user input will be run in 3 thread.
- ~MyWindow ()

7.5.1 Detailed Description

class that handle screen box, input box, data_structures box Finite state machine

Definition at line 23 of file DuckWin.hpp.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 MyWindow()

```
MyWindow::MyWindow ( )
```

contructor of MyWindow

Definition at line 5 of file DuckWin.cpp.

```
6 {
7     soundMutex.lock();
8     status = 0;
9     WIDTH = 0;
10     HEIGHT = 0;
11     window = nullptr;
12     renderer = nullptr;
13     FocusOn = 0;
14     screen.clear();
15     DT = nullptr;
16     input = nullptr;
17     turnOn = true;
18     soundOn = true;
19 }
```

7.5.2.2 ~MyWindow()

```
MyWindow::~MyWindow ( )
```

Definition at line 871 of file DuckWin.cpp.

```
872 {
873     shutdown();
874 }
```

7.5.3 Member Function Documentation

7.5.3.1 action()

```
void MyWindow::action ( )
```

get user input

Definition at line 990 of file DuckWin.cpp.

```
991 {
992     while(isOpen())
993     {
994         while(SDL_PollEvent(&event))
995             process();
996     }
997 }
```

7.5.3.2 changeScreens()

```
void MyWindow::changeScreens (
            const char *const & name )
```

Delete all screens and add new screens which information is in json file have directory is "GLOBAL::Atrb↩ Screens/name".

Definition at line 835 of file DuckWin.cpp.

```
836 {
837     deleteScreen();
838
839     json mem;
840
841     readjson(GLOBAL::AtrbScreens, name, mem);
842
843     screen.resize(mem.size());
844
845
846     for(int i = 0; i < (int) screen.size(); i++)
847     {
848         FocusOn = i;
849         top() = new Display;
850         top()->setRenderer(renderer);
851         top()->init(mem[i]);
852     }
853     if(turnOn)
854     {
855         screen[0]->hideButton(0);
856         screen[0]->showButton(1);
857     }else
858     {
859         screen[0]->hideButton(1);
860         screen[0]->showButton(0);
861     }
862     FocusOn = 0;
863 }
```

7.5.3.3 init()

```
void MyWindow::init ( )
```

init the window default size is 960x540

default fps is 60

default sound is on

default status is 1

init window, renderer, audio, image, font

Definition at line 33 of file DuckWin.cpp.

```
34 {
35     status = 1;
36     WIDTH = 960;
37     HEIGHT = 540;
38     RANDOM::init();
39
40     SDL_Init(SDL_INIT_VIDEO | SDL_INIT_AUDIO);
41     window = SDL_CreateWindow(
42             "Dr Duck",
43             SDL_WINDOWPOS_CENTERED,
44             SDL_WINDOWPOS_CENTERED,
45             WIDTH,
46             HEIGHT,
47             SDL_WINDOW_SHOWN
48             );
49
```

```
50      renderer = SDL_CreateRenderer(
51              window,
52              -1,
53              SDL_RENDERER_ACCELERATED
54              );
55
56      IMG_Init(IMG_INIT_PNG | IMG_INIT_JPG);
57      TTF_Init();
58
59      SDL_Rect rect;
60      rect.x = 0;
61      rect.y = 0;
62      rect.w = WIDTH;
63      rect.h = HEIGHT;
64      fps = 60;
65
66      SDL_RenderSetViewport(renderer, &rect);
67
68
69      audioSpec.freq = 44100;
70      audioSpec.format = AUDIO_S16SYS;
71      audioSpec.channels = 2;
72      audioSpec.samples = 1024;
73      SDL_OpenAudio(&audioSpec, NULL);
74      SDL_AudioSpec waveSpec;
75      SDL_LoadWAV((GLOBAL::SoundFolder + "quark.wav").c_str(), &waveSpec, &waveBuffer, &waveLength);
76  }
```

7.5.3.4 isClose()

```
bool MyWindow::isClose ( )
```

Return true if MyWindow is close, false otherwise.

Definition at line 828 of file DuckWin.cpp.

```
829 {
830     return status == 0;
831 }
```

7.5.3.5 isHanging()

```
bool MyWindow::isHanging ( )
```

Definition at line 821 of file DuckWin.cpp.

```
822 {
823     return status == 2;
824 }
```

7.5.3.6 isOpen()

```
bool MyWindow::isOpen ( )
```

Return true if MyWindow is open, false otherwise.

Definition at line 816 of file DuckWin.cpp.

```
817 {
818     return status == 1;
819 }
```

7.5.3.7 loadScreen()

```
void MyWindow::loadScreen (
            Display *& screen,
            const json & mem )
```

7.5.3.8 mouseMove()

```
void MyWindow::mouseMove (
            int x,
            int y )
```

when user move mouse, this function will be called

Parameters

x	x coordinate of mouse
y	y coordinate of mouse When mouse is move it will trigger the button, screen.

Definition at line 84 of file DuckWin.cpp.

```
85 {
86     for(int i = 0; i < (int) screen.size(); i++)
87     {
88         if(screen[i]->isVisible() && screen[i]->isLiesInside(x, y))
89         {
90             FocusOn = i;
91             screen[i]->changeFocus(x, y);
92             screen[i]->mouseMove(x, y);
93         }else if (!screen[i]->isVisible() && screen[i]->isLiesInside(x, y))
94         {
95             if(screen[i]->getAppear() == 0) continue;
96             else if(screen[i]->getAppear() == 1) screen[i]->appearFromBot(0.4);
97             else if(screen[i]->getAppear() == 2) screen[i]->appearFromRight(0.4);
98             FocusOn = i;
99             screen[i]->changeFocus(x, y);
100             screen[i]->mouseMove(x, y);
101         }else if(screen[i]->isVisible() && !screen[i]->isLiesInside(x, y))
102         {
103             if(screen[i]->getAppear() == 0) continue;
104             else if(screen[i]->getAppear() == 1) screen[i]->disappearToBot(0.4);
105             else if(screen[i]->getAppear() == 2) screen[i]->disappearToRight(0.4);
106         }
107     }
108     if(input != nullptr) input->mouseMove(x, y);
109 }
```

7.5.3.9 mousePress()

```
void MyWindow::mousePress (
            int x,
            int y )
```

When user press mouse, this function will be called to handle it.

Parameters

x	x coordinate of mouse
y	y coordinate of mouse

Definition at line 695 of file DuckWin.cpp.

```
696 {
697
698     if(input != nullptr) input->mousePress(x, y);
699
700     Button* but = nullptr;
701
702     if(input != nullptr)
703     {
704         but = input->getButtonPressedByMouse(x, y);
705     }
706     if(but == nullptr)
707     {
708         but = top()->mousePressedButton(x, y);
709     }
710     if(but == nullptr) return ;
711
712     if(but->getAction() == "setting")
713     {
714         UImutex.lock();
715         json mem;
716         readJson(GLOBAL::AtrbInputBox + "setting.json", mem);
717         if(input != nullptr) delete input;
718         input = new InputBox;
719         input->setRender(renderer);
720         input->init(mem);
721         UImutex.unlock();
722         return ;
723     }
724
725     if(but->getAction() == "sound on")
726     {
727         turnOn = true;
728         screen[0]->hideButton(0);
729         screen[0]->showButton(1);
730         return ;
731     }
732     if(but->getAction() == "sound off")
733     {
734         turnOn = false;
735         screen[0]->hideButton(1);
736         screen[0]->showButton(0);
737         return ;
738     }
739     soundOn = turnOn;
740     soundMutex.unlock();
741     if(soundOn) SDL_Delay(350);
742     else SDL_Delay(100);
743     soundMutex.lock();
744     if(isChangeScreen(but))
745     {
746         return ;
747     }
748     if(isDToperator(but))
749     {
750         return;
751     }
752     if(isInputButton(but))
753     {
754         return ;
755     }
756     if(isPlayButton(but))
757     {
758         return ;
759     }
760 }
```

7.5.3.10 process()

```
void MyWindow::process ( )
```

Handle user input.

Definition at line 941 of file DuckWin.cpp.

```
942 {
943     if(event.type == SDL_QUIT)
944     {
945         status = 0;
```

```
946         shutdown();
947     }else if(event.type == SDL_MOUSEMOTION)
948     {
949         mouseMove(event.motion.x, event.motion.y);
950     }else if(event.type == SDL_MOUSEBUTTONDOWN)
951     {
952         mousePress(event.motion.x, event.motion.y);
953         mouseMove(event.motion.x, event.motion.y);
954     } else if(event.type == SDL_KEYDOWN)
955     {
956         if(input != nullptr)
957         {
958             typing(event.key.keysym);
959         }
960     }
961 }
```

7.5.3.11 render()

```
void MyWindow::render ( )
```

render function

Definition at line 891 of file DuckWin.cpp.

```
892 {
893
894     double delayTime = 1000.0 / fps;
895     while(isOpen())
896     {
897         if(UImutex.try_lock())
898         {
899             Uint32 startTime = SDL_GetTicks();
900             SDL_RenderClear(renderer);
901
902             for(int i = 0; i < (int) screen.size(); i++)
903             {
904                 screen[i]->render();
905                 if(i == 0 && DT != nullptr)
906                 {
907                     DT->render();
908                 }
909             }
910             if(input != nullptr) input->render();
911             SDL_RenderPresent(renderer);
912             UImutex.unlock();
913             Uint32 DeltaTime = SDL_GetTicks() - startTime;
914             if(DeltaTime < delayTime)
915                 SDL_Delay(delayTime - delayTime);
916         }
917     }
918 }
```

7.5.3.12 run()

```
void MyWindow::run ( )
```

Start MyWindow render and sound, user input will be run in 3 thread.

Definition at line 879 of file DuckWin.cpp.

```
880 {
881     std::thread draw(&MyWindow::render, this);
882     std::thread sound(&MyWindow::speak, this);
883     action();
884
885     draw.join();
886     sound.join();
887 }
```

7.5.3.13 speak()

```
void MyWindow::speak ( )
```

sound function

Definition at line 922 of file DuckWin.cpp.

```
923 {
924     while(isOpen())
925     {
926         if(soundMutex.try_lock())
927         {
928             if(soundOn)
929             {
930                 SDL_QueueAudio(1, waveBuffer, waveLength);
931                 SDL_PauseAudio(0);
932             }
933             soundOn = false;
934             soundMutex.unlock();
935         }
936     }
937 }
```

7.5.3.14 top()

```
Display *& MyWindow::top ( )
```

Return the screen that is focus on.

Definition at line 867 of file DuckWin.cpp.

```
868 {
869     return screen[FocusOn];
870 }
```

7.5.3.15 typing()

```
void MyWindow::typing (
            SDL_Keysym key )
```

add a character to input box

Definition at line 965 of file DuckWin.cpp.

```
966 {
967     UImutex.lock();
968     if(key.sym == SDLK_BACKSPACE)
969         input->pop();
970     else if(key.sym == SDLK_TAB)
971         input->nextFocus();
972     else if(key.sym >= SDLK_a && key.sym <= SDLK_z)
973     {
974         char value = key.sym;
975         if(((SDL_GetModState() & KMOD_CAPS) != 0) ^ ((SDL_GetModState() & KMOD_SHIFT) != 0))
976             value = 'A' + (value - SDLK_a);
977         input->typing(value);
978     }else if(key.sym >= SDLK_SPACE && key.sym < SDLK_a)
979     {
980         char value = key.sym;
981         if(event.key.keysym.sym == SDLK_MINUS && (SDL_GetModState() & KMOD_SHIFT))
982             value = '_';
983         input->typing(value);
984     }
985     UImutex.unlock();
986 }
```

The documentation for this class was generated from the following files:

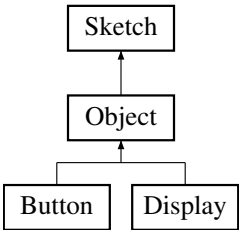
- include/DuckWin.hpp
- src/DuckWin.cpp

7.6 Object Class Reference

class that create an object and render it to the screen texture can be load from image or create new one with text and background color

`#include <Object.hpp>`

Inheritance diagram for Object:



Public Member Functions

- Object ()
  constructor of Object
- ~Object ()
- void clearTextures ()
  clear all textures
- void init (const json &mem, SDL_Renderer *&r)
  init this object by json file
- void setCoor (int x, int y, int w, int h)
  set coordinate of this object
- void setCoor (SDL_Rect key)
  set coordinate of this object
- void setX (int x)
  set x coordinate of this object
- void setY (int y)
  set y coordinate of this object
- void setW (int w)
  set width of this object
- void setH (int h)
  set height of this object
- const SDL_Rect & getCoor ()
  return coordinate of this object
- bool isLiesInside (int x, int y)
  return true if a point lies inside this object
- bool isLiesInside (int x, int y, int w, int h)
  return true if a rectangle inside this object
- bool isLiesInside (SDL_Rect rect)
  return true if a rectangle inside this object
- void addX (int k)
  add k to x coordinate
- void addY (int k)
  add k to y coordinate

- void addW (int k)

  add k to width
- void addH (int k)

  add k to height
- void show ()

  show this object
- void hide ()

  hide this object
- bool isVisible ()

  return true if this object is visible
- void setTextures (const json &mem)

  load textures from json file
- void pickTexure (int k)

  choose top texture
- int size ()

  return number of textures
- void render (bool update)

  render this object
- void moveTo (int x, int y, double time)

  move this object to (x, y) in time seconds
- void init (const json &mem)

  init sketch from json
- void setRender (SDL_Renderer ∗&r)

  set render
- void render ()

  render if sketch is hided, do nothing render sketch if renderer is nullptr, it may cause error, require setRender() before render() render background fisrt then render text
- void addChar (char ch)

  typing text
- void popChar ()

  erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create
- void setText (std::string s)

  set text to be s
- void setTextColor (int r, int g, int b)

  set text color to be (r, g, b)
- const std::string & getText ()

  return text
- void setColor (SDL_Color c)

  set background color to be c
- void setColor (int r, int g, int b)

  set background color to be (r, g, b)
- void setColor (int r, int g, int b, int a)

  set background color to be (r, g, b a)
- void setInCenterX ()

  align text texture align x coordinate of text texture to be in the center of the background texture
- void setOnLeftSideX ()

  align text texture align x coordinate of text texture to be in the left side of the background texture
- void setOnRightSideX ()

  align text texture align x coordinate of text texture to be in the right side of the background texture
- void setInCenterY ()

align text texture align y coordinate of text texture to be in the center of the background texture

- void setOnLeftSideY ()

  align text texture align y coordinate of text texture to be in the top side of the background texture

- void setOnRightSideY ()

  align text texture align y coordinate of text texture to be in the bottom side of the background texture

- void align ()

  align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

- void setBorder (int w, int r, int g, int b, int a)

  set border

- void setBorderColor (int r, int g, int b)

  set border color

- void FillWithColor ()

  fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

- void FillWithColor (SDL_Color c)

  fill background color with color C

- void highlight ()

  hightlight the sketch this function will change color of background to invert color

- void unHighlight ()

  unhightlight the sketch this function will change color of background to normal color

- bool isLieInside (int x, int y)

  determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

Protected Member Functions

- void clearTexture (int k)

  clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault

- void initColor (const json &mem)

  init color from json

- void initFont (const json &mem)

  init font from json

- void initBorder (const json &mem)

  init border from json

- void createTextTexture ()

  create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

7.6.1 Detailed Description

class that create an object and render it to the screen texture can be load from image or create new one with text and background color

Definition at line 14 of file Object.hpp.

7.6.2 Constructor & Destructor Documentation

7.6.2.1 Object()

```
Object::Object ( )
```

constructor of Object

Definition at line 7 of file Object.cpp.

```
8 {
9     setCoor(0, 0, 0, 0);
10     hide();
11     tes.clear();
12     ren = nullptr;
13     top = 0;
14 }
```

7.6.2.2 ~Object()

```
Object::~Object ( )
```

Definition at line 197 of file Object.cpp.

```
198 {
199     clearTextures();
200     setCoor(0, 0, 0, 0);
201     hide();
202     ren = nullptr;
203     top = 0;
204 }
```

7.6.3 Member Function Documentation

7.6.3.1 addChar()

```
void Sketch::addChar (
            char ch )  [inherited]
```

typing text

Parameters

ch	character that will be add to the end of the text add a character to the end of the text after that new text texture will be created

Definition at line 101 of file Sketch.cpp.

```
102 {
103     text = text + ch;
104     createTextTexture();
105 }
```

7.6.3.2 addH()

```
void Object::addH (
            int k )
```

add k to height

Definition at line 308 of file Object.cpp.

```
309 {
310     coor.h += k;
311 }
```

7.6.3.3 addW()

```
void Object::addW (
            int k )
```

add k to width

Definition at line 301 of file Object.cpp.

```
302 {
303     coor.w += k;
304 }
```

7.6.3.4 addX()

```
void Object::addX (
            int k )
```

add k to x coordinate

Definition at line 287 of file Object.cpp.

```
288 {
289     coor.x += k;
290 }
```

7.6.3.5 addY()

```
void Object::addY (
            int k )
```

add k to y coordinate

Definition at line 294 of file Object.cpp.

```
295 {
296     coor.y += k;
297 }
```

7.6.3.6 align()

```
void Sketch::align ( )  [inherited]
```

align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

Definition at line 761 of file Sketch.cpp.

```
762 {
763
764     if(textAlignX == 1) setOnLeftSideX();
765     if(textAlignX == 2) setInCenterX();
766     if(textAlignX == 3) setOnRightSideX();
767
768     if(textAlignY == 1) setOnLeftSideY();
769     if(textAlignY == 2) setInCenterY();
770     if(textAlignY == 3) setOnRightSideY();
771 }
```

7.6.3.7 clearTexture()

```
void Sketch::clearTexture (
            int k )  [protected], [inherited]
```

clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault

Parameters

k	integer, index of textures, 0 will be background, 1 will be text if tes[k] is nullptr, do nothing call SDL_DestroyTexture and after that set tes[k] to be nullptr

Definition at line 37 of file Sketch.cpp.

```
38 {
39     if(tes[k] == nullptr) return ;
40     SDL_DestroyTexture(tes[k]);
41     tes[k] = nullptr;
42 }
```

7.6.3.8 clearTextures()

```
void Object::clearTextures ( )
```

clear all textures

Definition at line 215 of file Object.cpp.

```
216 {
217     if(!tes.empty())
218     {
219         for(int i = 0; i < size(); i++)
220         {
221             if(tes[i] != nullptr)
222                 SDL_DestroyTexture(tes[i]);
223         }
224         tes.clear();
225     }
226 }
```

7.6.3.9 createTextTexture()

```
void Sketch::createTextTexture ( )  [protected], [inherited]
```

create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

Definition at line 63 of file Sketch.cpp.

```
64 {
65     clearTexture(1);
66     if(text.empty()) return ;
67 
68     SDL_Surface* surface = TTF_RenderText_Solid(font, text.c_str(), fontColor);
69 
70     tes[1] = SDL_CreateTextureFromSurface(ren, surface);
71 
72     coor[1].w = surface->w;
73     coor[1].h = surface->h;
74 
75     crop = coor[1];
76     crop.x = 0;
77     crop.y = 0;
78 
79     if(coor[1].w > coor[0].w || coor[1].h > coor[0].h)
80     {
81         crop = SDL_Rect({
82                 std::max(0, coor[1].w - coor[0].w),
83                 std::max(0, coor[1].h - coor[0].h),
84                 coor[0].w,
85                 coor[0].h
86             });
87         coor[1].w = coor[0].w;
88         coor[1].h = coor[0].h;
89     }
90 
91     align();
92 
93     SDL_FreeSurface(surface);
94 }
```

7.6.3.10 FillWithColor() [1/2]

```
void Sketch::FillWithColor ( )  [inherited]
```

fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

Definition at line 394 of file Sketch.cpp.

```
395 {
396     int w = coor[0].w;
397     int h = coor[0].h;
398     clearTexture(0);
399 
400     SDL_Surface* surf = SDL_CreateRGBSurfaceWithFormat(0, w, h, 32, SDL_PIXELFORMAT_RGBA32);
401     SDL_SetSurfaceBlendMode(surf, SDL_BLENDMODE_BLEND);
402 
403     SDL_FillRect(surf, nullptr, SDL_MapRGBA(surf->format, color.r, color.g, color.b, color.a));
404 
405     SDL_Rect borderRect;
406 
407     Uint32 c = SDL_MapRGBA(surf->format, borderColor.r, borderColor.g, borderColor.b, borderColor.a);
408     borderRect = SDL_Rect({0, 0, borderWidth, h});
409     SDL_FillRect(surf, &borderRect, c);
410 
411     borderRect = SDL_Rect({0, 0, w, borderWidth});
412     SDL_FillRect(surf, &borderRect, c);
413 
414     borderRect = SDL_Rect({0, h - borderWidth, w, borderWidth});
415     SDL_FillRect(surf, &borderRect, c);
416 
417     borderRect = SDL_Rect({w - borderWidth, 0, borderWidth, h});
418     SDL_FillRect(surf, &borderRect, c);
```

```
419
420     tes[0] = SDL_CreateTextureFromSurface(ren, surf);
421
422     SDL_FreeSurface(surf);
423
424 }
```

7.6.3.11 FillWithColor() [2/2]

```
void Sketch::FillWithColor (
            SDL_Color c )  [inherited]
```

fill background color with color C

Parameters

c	SDL_Color, color to fill fill background color with color C

Definition at line 381 of file Sketch.cpp.

```
382 {
383     SDL_Color temp = color;
384     color = c;
385     FillWithColor();
386     color = temp;
387 }
```

7.6.3.12 getCoor()

```
const SDL_Rect & Object::getCoor ( )
```

return coordinate of this object

Definition at line 133 of file Object.cpp.

```
134 {
135     return coor;
136 }
```

7.6.3.13 getText()

```
const std::string & Sketch::getText ( )  [inherited]
```

return text

Definition at line 148 of file Sketch.cpp.

```
149 {
150     return text;
151 }
```

7.6.3.14 hide()

```
void Object::hide ( )
```

hide this object

Definition at line 147 of file Object.cpp.

```
148 {
149     visable = false;
150 }
```

7.6.3.15 highlight()

```
void Sketch::highlight ( )  [inherited]
```

hightlight the sketch this function will change color of background to invert color

Definition at line 867 of file Sketch.cpp.

```
868 {
869     color.r = 255 - color.r;
870     color.g = 255 - color.g;
871     color.b = 255 - color.g;
872     if(color.r > 20 && color.g > 20 && color.b > 20)
873     {
874         color.r -= color.r * 0.3;
875         color.g -= color.g * 0.3;
876         color.b -= color.b * 0.3;
877     }
878     FillWithColor();
879 }
```

7.6.3.16 init() [1/2]

```
void Sketch::init (
            const json & mem )  [inherited]
```

init sketch from json

this function call initRect, initColor, initFont, initBorder this function also change visible, if mem contain "visible" key this function will fill with color, if mem contain "fill with color" key

Parameters

mem	json

example of param mem:

{

```
 "rect":
 {
```

```
 },
 "color":
 {
 },
 "font":
 {
 },
 "border":
 {
 },
 "text": "text",
 "visible": true,
"fill with color": true
```

}

Definition at line 738 of file Sketch.cpp.

```
739 {
740
741     initRect(mem);
742     initColor(mem);
743     initFont(mem);
744     initBorder(mem);
745
746     if(mem.contains("text"))
747     {
748         setText(mem["text"].get<std::string>());
749     }
750     if(mem.contains("visible"))
751         visible = mem["visible"];
752     if(mem.contains("fill with color"))
753     {
754         FillWithColor();
755     }
756 }
```

7.6.3.17 init() [2/2]

```
void Object::init (
             const json & mem,
             SDL_Renderer *& r )
```

init this object by json file

Parameters

mem	json file
r	renderer

Definition at line 63 of file Object.cpp.

```
64 {
65     ren = r;
```

```
66
67      Sketch::init(mem);
68
69      initTextures(mem);
70      initRect(mem);
71      initVisible(mem);
72  }
```

7.6.3.18 initBorder()

```
void Sketch::initBorder (
            const json & mem )  [protected], [inherited]
```

init border from json

if mem is not contain "border" key, do nothing

if in "border" object contain "width" key, set width of border to be mem["border"]["width"]

if in "border" object contain "color" key, set color of border to be mem["border"]["color"]

example of param mem:

{

"border": {

```
"width": 0,

"color": {

  "r": 0,

  "g": 0,

  "b": 0,

  "a": 0

}
```

}

}

Parameters

mem	json, contain border of sketch

Definition at line 667 of file Sketch.cpp.

```
668 {
669     if(!mem.contains("border")) return ;
670     if(mem["border"].contains("width"))
671         borderWidth = mem["border"]["width"];
672
673     if(mem["border"].contains("color"))
674     {
675         if(mem["border"]["color"].contains("r"))
676         {
```

```
677             borderColor.r = mem["border"]["color"]["r"];
678         }
679         if(mem["border"]["color"].contains("g"))
680         {
681             borderColor.g = mem["border"]["color"]["g"];
682         }
683         if(mem["border"]["color"].contains("b"))
684         {
685             borderColor.b = mem["border"]["color"]["b"];
686         }
687         if(mem["border"]["color"].contains("a"))
688         {
689             borderColor.a = mem["border"]["color"]["a"];
690         }
691     }
692 }
```

7.6.3.19 initColor()

```
void Sketch::initColor (
              const json & mem )  [protected], [inherited]
```

init color from json

if mem is not contain "color" key, do nothing

if in "color" object contain "r" key, set r color of sketch to be mem["color"]["r"]

if in "color" object contain "g" key, set g color of sketch to be mem["color"]["g"]

if in "color" object contain "b" key, set b color of sketch to be mem["color"]["b"]

if in "color" object contain "a" key, set a color of sketch to be mem["color"]["a"]

example of param mem:

{

"color": {

```
 "r": 0,
 "g": 0,
 "b": 0,
 "a": 0
```

}

}

Parameters

mem	json, contain color of sketch

Definition at line 512 of file Sketch.cpp.

```
513 {
```

```
514     if(mem.contains("color"))
515     {
516         if(mem["color"].contains("r"))
517             color.r = mem["color"]["r"];
518         if(mem["color"].contains("g"))
519             color.g = mem["color"]["g"];
520         if(mem["color"].contains("b"))
521             color.b = mem["color"]["b"];
522         if(mem["color"].contains("a"))
523             color.a = mem["color"]["a"];
524         cache = color;
525     }
526 }
```

7.6.3.20 initFont()

```
void Sketch::initFont (
            const json & mem )  [protected], [inherited]
```

init font from json

if mem is not contain "font" key, do nothing

get font file and combine with GLOBAL::FontsFolder to get full path of font file

source font from that path and source the size of the font

if in "font" object contain "rect" key, get rect text

if in "font" object contain "color" key, get color text

if int "font" object contain "text", set default text of sketch to be mem["font"]["text"]

example of param mem:

{

"font": {

```
 "name": "font.ttf",
 "size": 0,
 "rect": {
      "x": 0,
      "y": 0,
      "w": 0,
      "h": 0
},
"color": {
      "r": 0,
      "g": 0,
      "b": 0,
      "a": 0
},
"text": "text"
```

}

}

Parameters

mem	json, contain font of sketch

Definition at line 584 of file Sketch.cpp.

```
585 {
586     if(!mem.contains("font")) return ;
587     if(mem["font"].contains("name") && mem["font"].contains("size"))
588     {
589         char* name = combineLink(GLOBAL::FontsFolder, mem["font"]["name"].get<std::string>().c_str());
590         if(font != nullptr)
591         {
592             TTF_CloseFont(font);
593             font = nullptr;
594         }
595         font = TTF_OpenFont(name, mem["font"]["size"]);
596     }
597     if(mem["font"].contains("rect"))
598     {
599         if(mem["font"]["rect"].contains("x"))
600             coor[1].x = mem["font"]["rect"]["x"];
601         if(mem["font"]["rect"].contains("y"))
602             coor[1].y = mem["font"]["rect"]["y"];
603         if(mem["font"]["rect"].contains("align X"))
604             textAlignX = mem["font"]["rect"]["align X"];
605         if(mem["font"]["rect"].contains("align Y"))
606             textAlignX = mem["font"]["rect"]["align Y"];
607     }
608     if(mem["font"].contains("color"))
609     {
610         if(mem["font"]["color"].contains("r"))
611         {
612             fontColor.r = mem["font"]["color"]["r"];
613         }
614         if(mem["font"]["color"].contains("g"))
615         {
616             fontColor.g = mem["font"]["color"]["g"];
617         }
618         if(mem["font"]["color"].contains("b"))
619         {
620             fontColor.b = mem["font"]["color"]["b"];
621         }
622         if(mem["font"]["color"].contains("a"))
623         {
624             fontColor.a = mem["font"]["color"]["a"];
625         }
626     }
627     if(mem["font"].contains("text"))
628     {
629         setText(mem["font"]["text"].get<std::string>());
630     }
631 }
```

7.6.3.21 isLieInside()

```
bool Sketch::isLieInside (
            int x,
            int y )  [inherited]
```

determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

Parameters

x	int
y	int

Returns

bool

Definition at line 813 of file Sketch.cpp.

```
814 {
815     if(x < coor[0].x || coor[0].x + coor[0].w <= x) return false;
816     if(y < coor[0].y || coor[0].y + coor[0].y <= y) return false;
817     return true;
818 }
```

7.6.3.22 isLiesInside() [1/3]

```
bool Object::isLiesInside (
            int x,
            int y )
```

return true if a point lies inside this object

Definition at line 258 of file Object.cpp.

```
259 {
260     if(x < coor.x || coor.x + coor.w <= x)
261         return false;
262     if(y < coor.y || coor.y + coor.h <= y)
263         return false;
264     return true;
265 }
```

7.6.3.23 isLiesInside() [2/3]

```
bool Object::isLiesInside (
            int x,
            int y,
            int w,
            int h )
```

return true if a rectangle inside this object

Definition at line 269 of file Object.cpp.

```
270 {
271     if(x < coor.x || coor.x + coor.w <= x + w)
272         return false;
273     if(y < coor.y || coor.y + coor.h <= y + h)
274         return false;
275     return true;
276 }
```

7.6.3.24 isLiesInside() [3/3]

```
bool Object::isLiesInside (
            SDL_Rect rect )
```

return true if a rectangle inside this object

Definition at line 280 of file Object.cpp.

```
281 {
282     return isLiesInside(rect.x, rect.y, rect.w, rect.h);
283 }
```

7.6.3.25 isVisible()

```
bool Object::isVisible ( )
```

return true if this object is visible

Definition at line 250 of file Object.cpp.

```
251 {
252     return visable;
253 }
```

7.6.3.26 moveTo()

```
void Object::moveTo (
              int x,
              int y,
              double time )
```

move this object to (x, y) in time seconds

Definition at line 316 of file Object.cpp.

```
317 {
318     int dx = x - getCoor().x;
319     int dy = y - getCoor().y;
320
321     if(diff(time, 0))
322     {
323         coor.x = x;
324         coor.y = y;
325         return ;
326     }
327
328     double velo;
329     if(abs(dx) < abs(dy))
330         velo = dy / time;
331     else velo = dx / time;
332
333     int loop = std::min(80.0, abs(velo * time));
334
335     time = time / loop;
336
337
338     for(int i = 1; i <= loop; i++)
339     {
340         Uint32 startTime = SDL_GetTicks();
341
342         addX(-dx * (i - 1) / loop);
343         addX(dx * i / loop);
344         addY(-dy * (i - 1) / loop);
345         addY(dy * i / loop);
346         render(true);
347         Uint32 deltaTime = SDL_GetTicks() - startTime;
348         startTime = SDL_GetTicks();
349
350         if(deltaTime <= time * 1000)
351             SDL_Delay(time * 1000 - deltaTime);
352     }
353     setX(x);
354     setY(y);
355     render(true);
356 }
```

7.6.3.27 pickTexure()

```
void Object::pickTexure (
            int k )
```

choose top texture

Definition at line 231 of file Object.cpp.

```
232 {
233     if(k >= size()) return ;
234     top = k;
235 }
```

7.6.3.28 popChar()

```
void Sketch::popChar ( )  [inherited]
```

erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create

Definition at line 112 of file Sketch.cpp.

```
113 {
114     if(text.empty()) return ;
115     text.pop_back();
116     createTextTexture();
117 }
```

7.6.3.29 render() [1/2]

```
void Sketch::render ( )  [inherited]
```

render if sketch is hided, do nothing render sketch if renderer is nullptr, it may cause error, require setRender() before render() render background fisrt then render text

Definition at line 328 of file Sketch.cpp.

```
329 {
330     if(!isVisible()) return ;
331     if(tes[0] != nullptr) SDL_RenderCopy(ren, tes[0], nullptr, &coor[0]);
332     if(tes[1] != nullptr) SDL_RenderCopy(ren, tes[1], &crop, &coor[1]);
333 }
```

7.6.3.30 render() [2/2]

```
void Object::render (
            bool update )
```

render this object

Definition at line 239 of file Object.cpp.

```
240 {
241     if(!isVisible()) return ;
242     SDL_RenderCopy(ren, tes[top], nullptr, &coor);
243     if(update == true)
244         SDL_RenderPresent(ren);
245     return ;
246 }
```

7.6.3.31 setBorder()

```
void Sketch::setBorder (
            int w,
            int r,
            int g,
            int b,
            int a )  [inherited]
```

set border

set border of sketch

Parameters

w	interger, width of border
r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255
a	interger, alpha value of border color, 0 - 255

Definition at line 355 of file Sketch.cpp.

```
356 {
357     borderWidth = w;
358     borderColor.r = r;
359     borderColor.g = g;
360     borderColor.b = b;
361     borderColor.a = a;
362 }
```

7.6.3.32 setBorderColor()

```
void Sketch::setBorderColor (
            int r,
            int g,
            int b )  [inherited]
```

set border color

Parameters

r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255 set border color

Definition at line 370 of file Sketch.cpp.

```
371 {
372     borderColor.r = r;
373     borderColor.g = g;
374     borderColor.b = b;
375 }
```

7.6.3.33 setColor() [1/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b )  [inherited]
```

set background color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set background color to be (r, g, b)

Definition at line 167 of file Sketch.cpp.

```
168 {
169     color.r = r;
170     color.g = g;
171     color.b = b;
172 }
```

7.6.3.34 setColor() [2/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b,
            int a )  [inherited]
```

set background color to be (r, g, b a)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255
a	interger, alpha value, 0 - 255 set background color to be (r, g, b, a)

Definition at line 182 of file Sketch.cpp.

```
183 {
184     color.r = r;
185     color.g = g;
186     color.b = b;
187     color.a = a;
188 }
```

7.6.3.35 setColor() [3/3]

```
void Sketch::setColor (
            SDL_Color c )  [inherited]
```

set background color to be c

Parameters

c	SDL_Color, background color

Definition at line 156 of file Sketch.cpp.

```
157 {
158     color = c;
159 }
```

7.6.3.36 setCoor() [1/2]

```
void Object::setCoor (
              int x,
              int y,
              int w,
              int h )
```

set coordinate of this object

Parameters

x	x coordinate
y	y coordinate
w	width
h	height

Definition at line 80 of file Object.cpp.

```
81 {
82     coor.x = x;
83     coor.y = y;
84     coor.w = w;
85     coor.h = h;
86 }
```

7.6.3.37 setCoor() [2/2]

```
void Object::setCoor (
              SDL_Rect key )
```

set coordinate of this object

Parameters

key	SDL_Rect

Definition at line 91 of file Object.cpp.

```
92 {
93     coor.x = key.x;
94     coor.y = key.y;
95     coor.w = key.w;
```

```
96     coor.h = key.h;
97 }
```

7.6.3.38 setH()

```
void Object::setH (
              int h )
```

set height of this object

Parameters

h	height

Definition at line 126 of file Object.cpp.

```
127 {
128     coor.h = h;
129 }
```

7.6.3.39 setInCenterX()

```
void Sketch::setInCenterX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the center of the background texture

Definition at line 269 of file Sketch.cpp.

```
270 {
271     int x = coor[0].x;
272     int w = coor[0].w;
273     coor[1].x = x + (w - coor[1].w) / 2;
274 }
```

7.6.3.40 setInCenterY()

```
void Sketch::setInCenterY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the center of the background texture

Definition at line 279 of file Sketch.cpp.

```
280 {
281     int y = coor[0].y;
282     int h = coor[0].h;
283     coor[1].y = y + (h - coor[1].h) / 2;
284 }
```

7.6.3.41 setOnLeftSideX()

```
void Sketch::setOnLeftSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the left side of the background texture

Definition at line 289 of file Sketch.cpp.

```
290 {
291     coor[1].x = coor[0].x;
292 }
```

7.6.3.42 setOnLeftSideY()

```
void Sketch::setOnLeftSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the top side of the background texture

Definition at line 307 of file Sketch.cpp.

```
308 {
309     coor[1].y = coor[0].y;
310 }
```

7.6.3.43 setOnRightSideX()

```
void Sketch::setOnRightSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the right side of the background texture

Definition at line 297 of file Sketch.cpp.

```
298 {
299     int x = coor[0].x;
300     int w = coor[0].w;
301     coor[1].x = x + w - coor[1].w;
302 }
```

7.6.3.44 setOnRightSideY()

```
void Sketch::setOnRightSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the bottom side of the background texture

Definition at line 315 of file Sketch.cpp.

```
316 {
317     int y = coor[0].y;
318     int h = coor[0].h;
319     coor[1].y = y + h - coor[1].h;
320 }
```

7.6.3.45 setRender()

```
void Sketch::setRender (
             SDL_Renderer *& r )  [inherited]
```

set render

Parameters

r	address of SDL_Renderer pointer set render of sketch

Definition at line 339 of file Sketch.cpp.

```
340 {
341     ren = r;
342 }
```

7.6.3.46 setText()

```
void Sketch::setText (
            std::string s )  [inherited]
```

set text to be s

Parameters

s	string that will be set to text set text to be s and create new text texture

Definition at line 124 of file Sketch.cpp.

```
125 {
126     text = s;
127     createTextTexture();
128 }
```

7.6.3.47 setTextColor()

```
void Sketch::setTextColor (
            int r,
            int g,
            int b )  [inherited]
```

set text color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set text color to be (r, g, b) and create new text texture

Definition at line 136 of file Sketch.cpp.

```
137 {
138     fontColor.r = r;
139     fontColor.g = g;
140     fontColor.b = b;
141     createTextTexture();
142 }
```

7.6.3.48 setTextures()

```
void Object::setTextures (
            const json & mem )
```

load textures from json file

Definition at line 154 of file Object.cpp.

```
155 {
156     clearTextures();
157 
158     tes.resize(mem["textures"].size());
159 
160     char* FolderName = new char [256];
161     strcpy(FolderName, mem["name"].get<std::string>().c_str());
162 
163     for(int i = 0 ; i < size(); i++)
164     {
165         const char* fullname = combineName(
166             mem["textures"][i]["name"].get<std::string>().c_str(),
167             mem["textures"][i]["type"].get<std::string>().c_str()
168         );
169         const char* name = combineLink(
170             FolderName,
171             fullname
172         );
173         const char* link = combineLink(
174             GLOBAL::GraphicsFolder,
175             name
176         );
177 
178         SDL_Surface* surf;
179 
180         std::string type = mem["textures"][i]["type"].get<std::string>();
181         if(type == "bmp")
182             surf = SDL_LoadBMP(link);
183         else if(type == "png" || type == "jpg")
184             surf = IMG_Load(link);
185 
186         tes[i] = SDL_CreateTextureFromSurface(ren, surf);
187 
188         delete [] link;
189         delete [] name;
190         delete []fullname;
191         SDL_FreeSurface(surf);
192     }
193     delete [] FolderName;
194 }
```

7.6.3.49 setW()

```
void Object::setW (
            int w )
```

set width of this object

Parameters

w	width

Definition at line 118 of file Object.cpp.

```
119 {
120     coor.w = w;
121 }
```

7.6.3.50 setX()

```
void Object::setX (
            int x )
```

set x coordinate of this object

Parameters

x	x coordinate

Definition at line 102 of file Object.cpp.

```
103 {
104     coor.x = x;
105 }
```

7.6.3.51 setY()

```
void Object::setY (
            int y )
```

set y coordinate of this object

Parameters

y	y coordinate

Definition at line 110 of file Object.cpp.

```
111 {
112     coor.y = y;
113 }
```

7.6.3.52 show()

```
void Object::show ( )
```

show this object

Definition at line 140 of file Object.cpp.

```
141 {
142     visable = true;
143 }
```

7.6.3.53 size()

```
int Object::size ( )
```

return number of textures

Definition at line 208 of file Object.cpp.

```
209 {
210     return tes.size();
211 }
```

7.6.3.54 unHighlight()

```
void Sketch::unHighlight ( )  [inherited]
```

unhightlight the sketch this function will change color of background to normal color

Definition at line 884 of file Sketch.cpp.

```
885 {
886     color = cache;
887     FillWithColor();
888 }
```

The documentation for this class was generated from the following files:

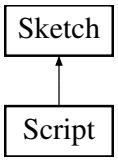
- include/Object.hpp
- src/Object.cpp

7.7 Script Class Reference

class that load an text image and render it to the screen Support highlight lines

```
#include <Script.hpp>
```

Inheritance diagram for Script:



Public Member Functions

- Script ()
- ~Script ()
- void loadObject (const json &mem)
- void loadHighlight (const json &mem)
- void init (const json &mem)
- void highlightLine (int k)
- void unHighlighLine (int k)
- void setRender (SDL_Renderer *&r)
- void render ()
- bool isVisible ()

  get visible this function will return visible of sketch
- void show ()

  show the sketch this function will set visible to true, that will enable the sketch to be rendered
- void hide ()

  hide the sketch this function will set visible to false, that will disable the sketch to be rendered
- void addChar (char ch)

  typing text
- void popChar ()

erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create

- void setText (std::string s)

  set text to be s
- void setTextColor (int r, int g, int b)

  set text color to be (r, g, b)
- const std::string & getText ()

  return text
- void setColor (SDL_Color c)

  set background color to be c
- void setColor (int r, int g, int b)

  set background color to be (r, g, b)
- void setColor (int r, int g, int b, int a)

  set background color to be (r, g, b a)
- void setCoor (int x, int y, int w, int h)

  set coordinate of sketch
- void setX (int x)

  set coordinate of sketch
- void setY (int y)

  set coordinate of sketch
- void setW (int w)

  set coordinate of sketch
- void setH (int h)

  set coordinate of sketch
- void addX (int x)

  set coordinate of sketch
- void addY (int y)

  set coordinate of sketch
- void setInCenterX ()

  align text texture align x coordinate of text texture to be in the center of the background texture
- void setOnLeftSideX ()

  align text texture align x coordinate of text texture to be in the left side of the background texture
- void setOnRightSideX ()

  align text texture align x coordinate of text texture to be in the right side of the background texture
- void setInCenterY ()

  align text texture align y coordinate of text texture to be in the center of the background texture
- void setOnLeftSideY ()

  align text texture align y coordinate of text texture to be in the top side of the background texture
- void setOnRightSideY ()

  align text texture align y coordinate of text texture to be in the bottom side of the background texture
- void align ()

  align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY
- SDL_Rect getCoor ()

  get coordinate this function will return coordinate of background of sketch
- void setBorder (int w, int r, int g, int b, int a)

  set border
- void setBorderColor (int r, int g, int b)

  set border color
- void FillWithColor ()

fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

- void FillWithColor (SDL_Color c)

  fill background color with color C
- void highlight ()

  hightlight the sketch this function will change color of background to invert color
- void unHighlight ()

  unhightlight the sketch this function will change color of background to normal color
- bool isLieInside (int x, int y)

  determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch
- void moveTo (int x, int y, double time)

  animation of sketch to move the sketch to point (x, y) in time (second) this function will move the sketch to point (x, y) in time (second)

Protected Member Functions

- void clearTexture (int k)

  clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault
- void initRect (const json &mem)

  set cooridnate of sketch from json
- void initColor (const json &mem)

  init color from json
- void initFont (const json &mem)

  init font from json
- void initBorder (const json &mem)

  init border from json
- void createTextTexture ()

  create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

7.7.1 Detailed Description

class that load an text image and render it to the screen Support highlight lines

Definition at line 14 of file Script.hpp.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 Script()

```
Script::Script ( )
```

Definition at line 3 of file Script.cpp.

```
4 {
5     obj = nullptr;
6     ren = nullptr;
7 }
```

7.7.2.2 ~Script()

```
Script::~Script ( )
```

Definition at line 10 of file Script.cpp.

```
11 {
12     for(int i = 0; i < (int)lines.size(); i++)
13         delete lines[i];
14     lines.clear();
15     if(obj != nullptr) delete obj;
16     obj = nullptr;
17     ren = nullptr;
18 }
```

7.7.3 Member Function Documentation

7.7.3.1 addChar()

```
void Sketch::addChar (
            char ch )  [inherited]
```

typing text

Parameters

ch	character that will be add to the end of the text add a character to the end of the text after that new text texture will be created

Definition at line 101 of file Sketch.cpp.

```
102 {
103     text = text + ch;
104     createTextTexture();
105 }
```

7.7.3.2 addX()

```
void Sketch::addX (
            int x )  [inherited]
```

set coordinate of sketch

Parameters

x,interger,change	of x coordinate of the top left corner of the sketch add x to x coordinate of sketch

Definition at line 218 of file Sketch.cpp.

```
219 {
220     coor[0].x += x;
221     align();
222 }
```

7.7.3.3 addY()

```
void Sketch::addY (
            int y )  [inherited]
```

set coordinate of sketch

Parameters

y	interger, change of y coordinate of the top left corner of the sketch add y to y coordinate of sketch

Definition at line 229 of file Sketch.cpp.

```
230 {
231     coor[0].y += y;
232     align();
233 }
```

7.7.3.4 align()

```
void Sketch::align ( )  [inherited]
```

align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

Definition at line 761 of file Sketch.cpp.

```
762 {
763
764     if(textAlignX == 1) setOnLeftSideX();
765     if(textAlignX == 2) setInCenterX();
766     if(textAlignX == 3) setOnRightSideX();
767
768     if(textAlignY == 1) setOnLeftSideY();
769     if(textAlignY == 2) setInCenterY();
770     if(textAlignY == 3) setOnRightSideY();
771 }
```

7.7.3.5 clearTexture()

```
void Sketch::clearTexture (
            int k )  [protected], [inherited]
```

clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault

Parameters

k	integer, index of textures, 0 will be background, 1 will be text if tes[k] is nullptr, do nothing call SDL_DestroyTexture and after that set tes[k] to be nullptr

Definition at line 37 of file Sketch.cpp.

```
38 {
39     if(tes[k] == nullptr) return ;
40     SDL_DestroyTexture(tes[k]);
41     tes[k] = nullptr;
42 }
```

7.7.3.6 createTextTexture()

```
void Sketch::createTextTexture ( )  [protected], [inherited]
```

create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

Definition at line 63 of file Sketch.cpp.

```
64 {
65     clearTexture(1);
66     if(text.empty()) return ;
67 
68     SDL_Surface* surface = TTF_RenderText_Solid(font, text.c_str(), fontColor);
69 
70     tes[1] = SDL_CreateTextureFromSurface(ren, surface);
71 
72     coor[1].w = surface->w;
73     coor[1].h = surface->h;
74 
75     crop = coor[1];
76     crop.x = 0;
77     crop.y = 0;
78 
79     if(coor[1].w > coor[0].w || coor[1].h > coor[0].h)
80     {
81         crop = SDL_Rect({
82                 std::max(0, coor[1].w - coor[0].w),
83                 std::max(0, coor[1].h - coor[0].h),
84                 coor[0].w,
85                 coor[0].h
86             });
87         coor[1].w = coor[0].w;
88         coor[1].h = coor[0].h;
89     }
90 
91     align();
92 
93     SDL_FreeSurface(surface);
94 }
```

7.7.3.7 FillWithColor() [1/2]

```
void Sketch::FillWithColor ( )  [inherited]
```

fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

Definition at line 394 of file Sketch.cpp.

```
395 {
396     int w = coor[0].w;
397     int h = coor[0].h;
398     clearTexture(0);
399 
400     SDL_Surface* surf = SDL_CreateRGBSurfaceWithFormat(0, w, h, 32, SDL_PIXELFORMAT_RGBA32);
401     SDL_SetSurfaceBlendMode(surf, SDL_BLENDMODE_BLEND);
402 
403     SDL_FillRect(surf, nullptr, SDL_MapRGBA(surf->format, color.r, color.g, color.b, color.a));
404 
405     SDL_Rect borderRect;
406 
```

```
407     Uint32 c = SDL_MapRGBA(surf->format, borderColor.r, borderColor.g, borderColor.b, borderColor.a);
408     borderRect = SDL_Rect({0, 0, borderWidth, h});
409     SDL_FillRect(surf, &borderRect, c);
410
411     borderRect = SDL_Rect({0, 0, w, borderWidth});
412     SDL_FillRect(surf, &borderRect, c);
413
414     borderRect = SDL_Rect({0, h - borderWidth, w, borderWidth});
415     SDL_FillRect(surf, &borderRect, c);
416
417     borderRect = SDL_Rect({w - borderWidth, 0, borderWidth, h});
418     SDL_FillRect(surf, &borderRect, c);
419
420     tes[0] = SDL_CreateTextureFromSurface(ren, surf);
421
422     SDL_FreeSurface(surf);
423
424 }
```

7.7.3.8 FillWithColor() [2/2]

```
void Sketch::FillWithColor (
            SDL_Color c )  [inherited]
```

fill background color with color C

Parameters

c	SDL_Color, color to fill fill background color with color C

Definition at line 381 of file Sketch.cpp.

```
382 {
383     SDL_Color temp = color;
384     color = c;
385     FillWithColor();
386     color = temp;
387 }
```

7.7.3.9 getCoor()

```
SDL_Rect Sketch::getCoor ( )  [inherited]
```

get coordinate this function will return coordinate of background of sketch

Returns

SDL_Rect

Definition at line 777 of file Sketch.cpp.

```
778 {
779     return coor[0];
780 }
```

7.7.3.10 getText()

```
const std::string & Sketch::getText ( )  [inherited]
```

return text

Definition at line 148 of file Sketch.cpp.

```
149 {
150     return text;
151 }
```

7.7.3.11 hide()

```
void Sketch::hide ( )  [inherited]
```

hide the sketch this function will set visible to false, that will disable the sketch to be rendered

Definition at line 802 of file Sketch.cpp.

```
803 {
804     visible = false;
805 }
```

7.7.3.12 highlight()

```
void Sketch::highlight ( )  [inherited]
```

hightlight the sketch this function will change color of background to invert color

Definition at line 867 of file Sketch.cpp.

```
868 {
869     color.r = 255 - color.r;
870     color.g = 255 - color.g;
871     color.b = 255 - color.g;
872     if(color.r > 20 && color.g > 20 && color.b > 20)
873     {
874         color.r -= color.r * 0.3;
875         color.g -= color.g * 0.3;
876         color.b -= color.b * 0.3;
877     }
878     FillWithColor();
879 }
```

7.7.3.13 highlightLine()

```
void Script::highlightLine (
             int k )
```

Definition at line 58 of file Script.cpp.

```
59 {
60     lines[k]->show();
61 }
```

7.7.3.14 init()

```
void Script::init (
            const json & mem )
```

Definition at line 44 of file Script.cpp.

```
45 {
46     Sketch::setRender(ren);
47     Sketch::init(mem);
48     if(mem.contains("object"))
49     {
50         loadObject(mem["object"]);
51     }
52     if(mem.contains("highlight"))
53     {
54         loadHighlight(mem["highlight"]);
55     }
56 }
```

7.7.3.15 initBorder()

```
void Sketch::initBorder (
            const json & mem )  [protected], [inherited]
```

init border from json

if mem is not contain "border" key, do nothing

if in "border" object contain "width" key, set width of border to be mem["border"]["width"]

if in "border" object contain "color" key, set color of border to be mem["border"]["color"]

example of param mem:

{

"border": {

```
"width": 0,
"color": {
  "r": 0,
  "g": 0,
  "b": 0,
  "a": 0
}
```

}

}

Parameters

mem	json, contain border of sketch

Definition at line 667 of file Sketch.cpp.

```
668 {
669     if(!mem.contains("border")) return ;
670     if(mem["border"].contains("width"))
671         borderWidth = mem["border"]["width"];
672 
673     if(mem["border"].contains("color"))
674     {
675         if(mem["border"]["color"].contains("r"))
676         {
677             borderColor.r = mem["border"]["color"]["r"];
678         }
679         if(mem["border"]["color"].contains("g"))
680         {
681             borderColor.g = mem["border"]["color"]["g"];
682         }
683         if(mem["border"]["color"].contains("b"))
684         {
685             borderColor.b = mem["border"]["color"]["b"];
686         }
687         if(mem["border"]["color"].contains("a"))
688         {
689             borderColor.a = mem["border"]["color"]["a"];
690         }
691     }
692 }
```

7.7.3.16 initColor()

```
void Sketch::initColor (
            const json & mem )  [protected], [inherited]
```

init color from json

if mem is not contain "color" key, do nothing

if in "color" object contain "r" key, set r color of sketch to be mem["color"]["r"]

if in "color" object contain "g" key, set g color of sketch to be mem["color"]["g"]

if in "color" object contain "b" key, set b color of sketch to be mem["color"]["b"]

if in "color" object contain "a" key, set a color of sketch to be mem["color"]["a"]

example of param mem:

{

"color": {

```
 "r": 0,
 "g": 0,
 "b": 0,
 "a": 0
```

}

}

Parameters

mem	json, contain color of sketch

Definition at line 512 of file Sketch.cpp.

```
513 {
514     if(mem.contains("color"))
515     {
516         if(mem["color"].contains("r"))
517             color.r = mem["color"]["r"];
518         if(mem["color"].contains("g"))
519             color.g = mem["color"]["g"];
520         if(mem["color"].contains("b"))
521             color.b = mem["color"]["b"];
522         if(mem["color"].contains("a"))
523             color.a = mem["color"]["a"];
524         cache = color;
525     }
526 }
```

7.7.3.17 initFont()

```
void Sketch::initFont (
            const json & mem )  [protected], [inherited]
```

init font from json

if mem is not contain "font" key, do nothing

get font file and combine with GLOBAL::FontsFolder to get full path of font file

source font from that path and source the size of the font

if in "font" object contain "rect" key, get rect text

if in "font" object contain "color" key, get color text

if int "font" object contain "text", set default text of sketch to be mem["font"]["text"]

example of param mem:

{

"font": {

```
 "name": "font.ttf",
 "size": 0,
 "rect": {
      "x": 0,
      "y": 0,
      "w": 0,
      "h": 0
},
"color": {
```

```
      "r": 0,

      "g": 0,

      "b": 0,

      "a": 0

},

"text": "text"


}

}
```

Parameters

mem	json, contain font of sketch

Definition at line 584 of file Sketch.cpp.

```
585 {
586     if(!mem.contains("font")) return ;
587     if(mem["font"].contains("name") && mem["font"].contains("size"))
588     {
589         char* name = combineLink(GLOBAL::FontsFolder, mem["font"]["name"].get<std::string>().c_str());
590         if(font != nullptr)
591         {
592             TTF_CloseFont(font);
593             font = nullptr;
594         }
595         font = TTF_OpenFont(name, mem["font"]["size"]);
596     }
597     if(mem["font"].contains("rect"))
598     {
599         if(mem["font"]["rect"].contains("x"))
600             coor[1].x = mem["font"]["rect"]["x"];
601         if(mem["font"]["rect"].contains("y"))
602             coor[1].y = mem["font"]["rect"]["y"];
603         if(mem["font"]["rect"].contains("align X"))
604             textAlignX = mem["font"]["rect"]["align X"];
605         if(mem["font"]["rect"].contains("align Y"))
606             textAlignX = mem["font"]["rect"]["align Y"];
607     }
608     if(mem["font"].contains("color"))
609     {
610         if(mem["font"]["color"].contains("r"))
611         {
612             fontColor.r = mem["font"]["color"]["r"];
613         }
614         if(mem["font"]["color"].contains("g"))
615         {
616             fontColor.g = mem["font"]["color"]["g"];
617         }
618         if(mem["font"]["color"].contains("b"))
619         {
620             fontColor.b = mem["font"]["color"]["b"];
621         }
622         if(mem["font"]["color"].contains("a"))
623         {
624             fontColor.a = mem["font"]["color"]["a"];
625         }
626     }
627     if(mem["font"].contains("text"))
628     {
629         setText(mem["font"]["text"].get<std::string>());
630     }
631 }
```

7.7.3.18 initRect()

```
void Sketch::initRect (
```

```
            const json & mem )  [protected], [inherited]
```

set cooridnate of sketch from json

if mem is not contain "rect" key, do nothing

if in "rect" object contain "x" key, set x coordinate of sketch to be mem["rect"]["x"]

if in "rect" object contain "y" key, set y coordinate of sketch to be mem["rect"]["y"]

if in "rect" object contain "w" key, set w coordinate of sketch to be mem["rect"]["w"]

if in "rect" object contain "h" key, set h coordinate of sketch to be mem["rect"]["h"]

example of param mem:

{

```
"rect": {
     "x": 0,
     "y": 0,
     "w": 0,
     "h": 0
}
```

Parameters

mem	json, contain coordinate of sketch

Definition at line 457 of file Sketch.cpp.

```
458 {
459     if(mem.contains("rect"))
460     {
461         if(mem["rect"].contains("x"))
462         {
463             coor[0].x = mem["rect"]["x"];
464         }
465         if(mem["rect"].contains("y"))
466         {
467             coor[0].y = mem["rect"]["y"];
468         }
469         if(mem["rect"].contains("w"))
470         {
471             coor[0].w = mem["rect"]["w"];
472         }
473         if(mem["rect"].contains("h"))
474         {
475             coor[0].h = mem["rect"]["h"];
476         }
477     }
478 }
```

7.7.3.19 isLieInside()

```
bool Sketch::isLieInside (
            int x,
            int y )  [inherited]
```

determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

Parameters

x	int
y	int

Returns

bool

Definition at line 813 of file Sketch.cpp.

```
814 {
815     if(x < coor[0].x || coor[0].x + coor[0].w <= x) return false;
816     if(y < coor[0].y || coor[0].y + coor[0].y <= y) return false;
817     return true;
818 }
```

7.7.3.20 isVisible()

```
bool Sketch::isVisible ( )  [inherited]
```

get visible this function will return visible of sketch

Returns

bool

Definition at line 786 of file Sketch.cpp.

```
787 {
788     return visible;
789 }
```

7.7.3.21 loadHighlight()

```
void Script::loadHighlight (
            const json & mem )
```

Definition at line 28 of file Script.cpp.

```
29 {
30     lines.clear();
31     lines.resize(mem["size"]);
32     int dx = mem["dx"];
33     int dy = mem["dy"];
34 
35     for(int i = 0; i < (int) lines.size(); i++)
36     {
37         lines[i] = new Sketch;
38         lines[i]->setRender(ren);
39         lines[i]->init(mem);
40         lines[i]->addX(dx * i);
41         lines[i]->addY(dy * i);
42     }
43 }
```

7.7.3.22 loadObject()

```
void Script::loadObject (
            const json & mem )
```

Definition at line 20 of file Script.cpp.

```
21 {
22     if(obj != nullptr) delete obj;
23
24     obj = new Object;
25     obj->init(mem, ren);
26 }
```

7.7.3.23 moveTo()

```
void Sketch::moveTo (
            int x,
            int y,
            double time )  [inherited]
```

animation of sketch to move the sketch to point (x, y) in time (second) this function will move the sketch to point (x, y) in time (second)

Parameters

x	int
y	int
time	double

Definition at line 826 of file Sketch.cpp.

```
827 {
828     int dx = x - getCoor().x;
829     int dy = y - getCoor().y;
830
831     if(diff(time, 0))
832     {
833         setX(x);
834         setY(y);
835         return ;
836     }
837
838     double velo;
839
840     if(abs(dx) < abs(dy))
841         velo = dy / time;
842     else velo = dx / time;
843
844     int loop = std::min(80.0, abs(velo * time));
845
846     time = time / loop;
847
848     for(int i = 0; i <= loop; i++)
849     {
850         Uint32 startTime = SDL_GetTicks();
851
852         addX(-dx * (i - 1) / loop);
853         addX(dx * i / loop);
854         addY(-dy * (i - 1) / loop);
855         addY(dy * i / loop);
856         render();
857         Uint32 deltaTime = SDL_GetTicks() - startTime;
858         startTime = SDL_GetTicks();
859         if(deltaTime <= time * 1000)
860             SDL_Delay(time * 1000 - deltaTime);
861     }
862 }
```

7.7.3.24 popChar()

```
void Sketch::popChar ( )  [inherited]
```

erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create

Definition at line 112 of file Sketch.cpp.

```
113 {
114     if(text.empty()) return ;
115     text.pop_back();
116     createTextTexture();
117 }
```

7.7.3.25 render()

```
void Script::render ( )
```

Definition at line 72 of file Script.cpp.

```
73 {
74     if(!isVisible()) return ;
75     Sketch::render();
76     if(obj != nullptr) obj->render(false);
77     for(int i = 0; i < lines.size(); i++)
78         lines[i]->render();
79 }
```

7.7.3.26 setBorder()

```
void Sketch::setBorder (
            int w,
            int r,
            int g,
            int b,
            int a )  [inherited]
```

set border

set border of sketch

Parameters

w	interger, width of border
r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255
a	interger, alpha value of border color, 0 - 255

Definition at line 355 of file Sketch.cpp.

```
356 {
357     borderWidth = w;
358     borderColor.r = r;
359     borderColor.g = g;
360     borderColor.b = b;
361     borderColor.a = a;
362 }
```

7.7.3.27 setBorderColor()

```
void Sketch::setBorderColor (
            int r,
            int g,
            int b )  [inherited]
```

set border color

Parameters

r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255 set border color

Definition at line 370 of file Sketch.cpp.

```
371 {
372     borderColor.r = r;
373     borderColor.g = g;
374     borderColor.b = b;
375 }
```

7.7.3.28 setColor() [1/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b )  [inherited]
```

set background color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set background color to be (r, g, b)

Definition at line 167 of file Sketch.cpp.

```
168 {
169     color.r = r;
170     color.g = g;
171     color.b = b;
172 }
```

7.7.3.29 setColor() [2/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b,
            int a )  [inherited]
```

set background color to be (r, g, b a)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255
a	interger, alpha value, 0 - 255 set background color to be (r, g, b, a)

Definition at line 182 of file Sketch.cpp.

```
183 {
184     color.r = r;
185     color.g = g;
186     color.b = b;
187     color.a = a;
188 }
```

7.7.3.30 setColor() [3/3]

```
void Sketch::setColor (
            SDL_Color c )  [inherited]
```

set background color to be c

Parameters

c	SDL_Color, background color

Definition at line 156 of file Sketch.cpp.

```
157 {
158     color = c;
159 }
```

7.7.3.31 setCoor()

```
void Sketch::setCoor (
            int x,
            int y,
            int w,
            int h )  [inherited]
```

set coordinate of sketch

Parameters

x	interger, x coordinate of the top left corner of the sketch
y	interger, y coordinate of the top left corner of the sketch
w	interger, width of the sketch
h	interger, height of the sketch set coordinate of sketch

Definition at line 198 of file Sketch.cpp.

```
199 {
200     coor[0] = SDL_Rect({x, y, w, h});
201     align();
202 }
```

7.7.3.32 setH()

```
void Sketch::setH (
            int h )  [inherited]
```

set coordinate of sketch

Parameters

h	interger, height of the sketch set height of sketch

Definition at line 260 of file Sketch.cpp.

```
261 {
262     coor[0].h = h;
263     align();
264 }
```

7.7.3.33 setInCenterX()

```
void Sketch::setInCenterX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the center of the background texture

Definition at line 269 of file Sketch.cpp.

```
270 {
271     int x = coor[0].x;
272     int w = coor[0].w;
273     coor[1].x = x + (w - coor[1].w) / 2;
274 }
```

7.7.3.34 setInCenterY()

```
void Sketch::setInCenterY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the center of the background texture

Definition at line 279 of file Sketch.cpp.

```
280 {
281     int y = coor[0].y;
282     int h = coor[0].h;
283     coor[1].y = y + (h - coor[1].h) / 2;
284 }
```

7.7.3.35 setOnLeftSideX()

```
void Sketch::setOnLeftSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the left side of the background texture

Definition at line 289 of file Sketch.cpp.

```
290 {
291     coor[1].x = coor[0].x;
292 }
```

7.7.3.36 setOnLeftSideY()

```
void Sketch::setOnLeftSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the top side of the background texture

Definition at line 307 of file Sketch.cpp.

```
308 {
309     coor[1].y = coor[0].y;
310 }
```

7.7.3.37 setOnRightSideX()

```
void Sketch::setOnRightSideX ( )  [inherited]
```

align text texture align x coordinate of text texture to be in the right side of the background texture

Definition at line 297 of file Sketch.cpp.

```
298 {
299     int x = coor[0].x;
300     int w = coor[0].w;
301     coor[1].x = x + w - coor[1].w;
302 }
```

7.7.3.38 setOnRightSideY()

```
void Sketch::setOnRightSideY ( )  [inherited]
```

align text texture align y coordinate of text texture to be in the bottom side of the background texture

Definition at line 315 of file Sketch.cpp.

```
316 {
317     int y = coor[0].y;
318     int h = coor[0].h;
319     coor[1].y = y + h - coor[1].h;
320 }
```

7.7.3.39 setRender()

```
void Script::setRender (
            SDL_Renderer *& r )
```

Definition at line 67 of file Script.cpp.

```
68 {
69     ren = r;
70 }
```

7.7.3.40 setText()

```
void Sketch::setText (
            std::string s )  [inherited]
```

set text to be s

Parameters

s	string that will be set to text set text to be s and create new text texture

Definition at line 124 of file Sketch.cpp.

```
125 {
126     text = s;
127     createTextTexture();
128 }
```

7.7.3.41 setTextColor()

```
void Sketch::setTextColor (
            int r,
            int g,
            int b )  [inherited]
```

set text color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set text color to be (r, g, b) and create new text texture

Definition at line 136 of file Sketch.cpp.

```
137 {
138     fontColor.r = r;
139     fontColor.g = g;
140     fontColor.b = b;
141     createTextTexture();
142 }
```

7.7.3.42 setW()

```
void Sketch::setW (
            int w )  [inherited]
```

set coordinate of sketch

Parameters

w	interger, width of the sketch set width of sketch

Definition at line 250 of file Sketch.cpp.

```
251 {
252     coor[0].w = w;
253     align();
254 }
```

7.7.3.43 setX()

```
void Sketch::setX (
            int x )  [inherited]
```

set coordinate of sketch

Parameters

x	interger, x coordinate of the top left corner of the sketch set x coordinate of sketch

Definition at line 208 of file Sketch.cpp.

```
209 {
210     coor[0].x = x;
211     align();
212 }
```

7.7.3.44 setY()

```
void Sketch::setY (
            int y )  [inherited]
```

set coordinate of sketch

Parameters

y	interger, y coordinate of the top left corner of the sketch set y coordinate of sketch

Definition at line 240 of file Sketch.cpp.

```
241 {
242     coor[0].y = y;
243     align();
244 }
```

7.7.3.45 show()

```
void Sketch::show ( )  [inherited]
```

show the sketch this function will set visible to true, that will enable the sketch to be rendered

Definition at line 794 of file Sketch.cpp.

```
795 {
796     visible = true;
797 }
```

7.7.3.46 unHighlighLine()

```
void Script::unHighlighLine (
              int k )
```

Definition at line 62 of file Script.cpp.

```
63 {
64     lines[k]->hide();
65 }
```

7.7.3.47 unHighlight()

```
void Sketch::unHighlight ( )  [inherited]
```

unhightlight the sketch this function will change color of background to normal color

Definition at line 884 of file Sketch.cpp.

```
885 {
886     color = cache;
887     FillWithColor();
888 }
```

The documentation for this class was generated from the following files:

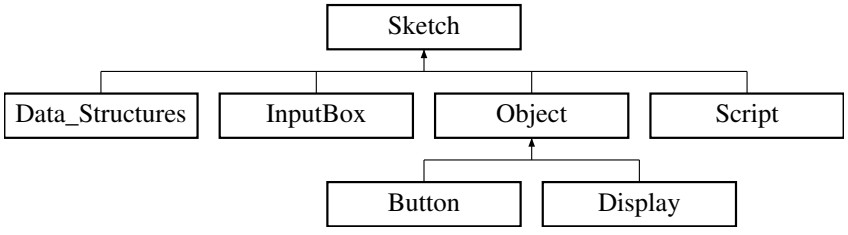
- include/Script.hpp
- src/Script.cpp

7.8 Sketch Class Reference

class that create an text box and render it to the screen

```
#include <Sketch.hpp>
```

Inheritance diagram for Sketch:



Public Member Functions

- Sketch ()

  sketch constructor
- ~Sketch ()

  destructor font and ren will not be deleted because it just a copy of the pointer call clearTexture(0) and clearTexture(1) to delete texture
- bool isVisible ()

  get visible this function will return visible of sketch
- void show ()

  show the sketch this function will set visible to true, that will enable the sketch to be rendered
- void hide ()

  hide the sketch this function will set visible to false, that will disable the sketch to be rendered
- void init (const json &mem)

  init sketch from json
- void setRender (SDL_Renderer *&r)

  set render
- void render ()

  render if sketch is hided, do nothing render sketch if renderer is nullptr, it may cause error, require setRender() before render() render background fisrt then render text
- void addChar (char ch)

  typing text
- void popChar ()

  erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create
- void setText (std::string s)

  set text to be s
- void setTextColor (int r, int g, int b)

  set text color to be (r, g, b)
- const std::string & getText ()

  return text
- void setColor (SDL_Color c)

  set background color to be c
- void setColor (int r, int g, int b)

  set background color to be (r, g, b)
- void setColor (int r, int g, int b, int a)

  set background color to be (r, g, b a)
- void setCoor (int x, int y, int w, int h)

  set coordinate of sketch
- void setX (int x)

  set coordinate of sketch
- void setY (int y)

  set coordinate of sketch
- void setW (int w)

  set coordinate of sketch
- void setH (int h)

  set coordinate of sketch
- void addX (int x)

  set coordinate of sketch
- void addY (int y)

  set coordinate of sketch
- void setInCenterX ()

align text texture align x coordinate of text texture to be in the center of the background texture

- void setOnLeftSideX ()

  align text texture align x coordinate of text texture to be in the left side of the background texture
- void setOnRightSideX ()

  align text texture align x coordinate of text texture to be in the right side of the background texture
- void setInCenterY ()

  align text texture align y coordinate of text texture to be in the center of the background texture
- void setOnLeftSideY ()

  align text texture align y coordinate of text texture to be in the top side of the background texture
- void setOnRightSideY ()

  align text texture align y coordinate of text texture to be in the bottom side of the background texture
- void align ()

  align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY
- SDL_Rect getCoor ()

  get coordinate this function will return coordinate of background of sketch
- void setBorder (int w, int r, int g, int b, int a)

  set border
- void setBorderColor (int r, int g, int b)

  set border color
- void FillWithColor ()

  fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black
- void FillWithColor (SDL_Color c)

  fill background color with color C
- void highlight ()

  hightlight the sketch this function will change color of background to invert color
- void unHighlight ()

  unhightlight the sketch this function will change color of background to normal color
- bool isLieInside (int x, int y)

  determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch
- void moveTo (int x, int y, double time)

  animation of sketch to move the sketch to point (x, y) in time (second) this function will move the sketch to point (x, y) in time (second)

Protected Member Functions

- void clearTexture (int k)

  clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault
- void initRect (const json &mem)

  set cooridnate of sketch from json
- void initColor (const json &mem)

  init color from json
- void initFont (const json &mem)

  init font from json
- void initBorder (const json &mem)

  init border from json
- void createTextTexture ()

  create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

7.8.1 Detailed Description

class that create an text box and render it to the screen

Definition at line 15 of file Sketch.hpp.

7.8.2 Constructor & Destructor Documentation

7.8.2.1 Sketch()

```
Sketch::Sketch ( )
```

sketch constructor

set all pointer to be nullptr default background color is black default font color is white and there will be no text default text align is center default border size is 0 default coordinate will be top left corner and size is 0

Definition at line 12 of file Sketch.cpp.

```
13 {
14     font = nullptr;
15     text = "";
16     color = SDL_Color({0, 0, 0, 255});
17     fontColor = SDL_Color({255, 255, 255, 255});
18 
19     tes[0] = nullptr;
20     tes[1] = nullptr;
21     coor[0] = SDL_Rect({0, 0, 0, 0});
22     coor[1] = coor[0];
23     ren = nullptr;
24 
25     borderWidth = 0;
26 
27     textAlignX = 2;
28     textAlignY = 2;
29 }
```

7.8.2.2 ~Sketch()

```
Sketch::~Sketch ( )
```

destructor font and ren will not be deleted because it just a copy of the pointer call clearTexture(0) and clear↩ Texture(1) to delete texture

Definition at line 48 of file Sketch.cpp.

```
49 {
50     font = nullptr;
51     text.clear();
52     clearTexture(0);
53     clearTexture(1);
54     ren = nullptr;
55 }
```

7.8.3 Member Function Documentation

7.8.3.1 addChar()

```
void Sketch::addChar (
            char ch )
```

typing text

Parameters

ch	character that will be add to the end of the text add a character to the end of the text after that new text texture will be created

Definition at line 101 of file Sketch.cpp.

```
102 {
103     text = text + ch;
104     createTextTexture();
105 }
```

7.8.3.2 addX()

```
void Sketch::addX (
            int x )
```

set coordinate of sketch

Parameters

x,interger,change	of x coordinate of the top left corner of the sketch add x to x coordinate of sketch

Definition at line 218 of file Sketch.cpp.

```
219 {
220     coor[0].x += x;
221     align();
222 }
```

7.8.3.3 addY()

```
void Sketch::addY (
            int y )
```

set coordinate of sketch

Parameters

y	interger, change of y coordinate of the top left corner of the sketch add y to y coordinate of sketch

Definition at line 229 of file Sketch.cpp.

```
230 {
231     coor[0].y += y;
232     align();
233 }
```

7.8.3.4 align()

```
void Sketch::align ( )
```

align text this function will call setOnLeftSideX, setOnRightSideX, setInCenterX, setOnLeftSideY, setOnRightSideY, setInCenterY

Definition at line 761 of file Sketch.cpp.

```
762 {
763
764     if(textAlignX == 1) setOnLeftSideX();
765     if(textAlignX == 2) setInCenterX();
766     if(textAlignX == 3) setOnRightSideX();
767
768     if(textAlignY == 1) setOnLeftSideY();
769     if(textAlignY == 2) setInCenterY();
770     if(textAlignY == 3) setOnRightSideY();
771 }
```

7.8.3.5 clearTexture()

```
void Sketch::clearTexture (
            int k )  [protected]
```

clear texture, k = 0 - background, k = 1 - text, anything else will cause segment fault

Parameters

k	integer, index of textures, 0 will be background, 1 will be text if tes[k] is nullptr, do nothing call SDL_DestroyTexture and after that set tes[k] to be nullptr

Definition at line 37 of file Sketch.cpp.

```
38 {
39     if(tes[k] == nullptr) return ;
40     SDL_DestroyTexture(tes[k]);
41     tes[k] = nullptr;
42 }
```

7.8.3.6 createTextTexture()

```
void Sketch::createTextTexture ( )  [protected]
```

create text texture delete old text texture if exist if text is empty then do nothing make sure that font is not nullptr, otherwise it may cause segment fault. if text texture is greater than background texture, crop it, the top left.

Definition at line 63 of file Sketch.cpp.

```
64 {
65     clearTexture(1);
66     if(text.empty()) return ;
67
68     SDL_Surface* surface = TTF_RenderText_Solid(font, text.c_str(), fontColor);
69
70     tes[1] = SDL_CreateTextureFromSurface(ren, surface);
71
72     coor[1].w = surface->w;
73     coor[1].h = surface->h;
74
75     crop = coor[1];
76     crop.x = 0;
77     crop.y = 0;
78
79     if(coor[1].w > coor[0].w || coor[1].h > coor[0].h)
80     {
81         crop = SDL_Rect({
```

```
82                 std::max(0, coor[1].w - coor[0].w),
83                 std::max(0, coor[1].h - coor[0].h),
84                 coor[0].w,
85                 coor[0].h
86             });
87         coor[1].w = coor[0].w;
88         coor[1].h = coor[0].h;
89     }
90
91     align();
92
93     SDL_FreeSurface(surface);
94 }
```

7.8.3.7 FillWithColor() [1/2]

```
void Sketch::FillWithColor ( )
```

fill background color with default color, which is set by SetColor function fill background color with default color, which is set by SetColor function at default color is black

Definition at line 394 of file Sketch.cpp.

```
395 {
396     int w = coor[0].w;
397     int h = coor[0].h;
398     clearTexture(0);
399
400     SDL_Surface* surf = SDL_CreateRGBSurfaceWithFormat(0, w, h, 32, SDL_PIXELFORMAT_RGBA32);
401     SDL_SetSurfaceBlendMode(surf, SDL_BLENDMODE_BLEND);
402
403     SDL_FillRect(surf, nullptr, SDL_MapRGBA(surf->format, color.r, color.g, color.b, color.a));
404
405     SDL_Rect borderRect;
406
407     Uint32 c = SDL_MapRGBA(surf->format, borderColor.r, borderColor.g, borderColor.b, borderColor.a);
408     borderRect = SDL_Rect({0, 0, borderWidth, h});
409     SDL_FillRect(surf, &borderRect, c);
410
411     borderRect = SDL_Rect({0, 0, w, borderWidth});
412     SDL_FillRect(surf, &borderRect, c);
413
414     borderRect = SDL_Rect({0, h - borderWidth, w, borderWidth});
415     SDL_FillRect(surf, &borderRect, c);
416
417     borderRect = SDL_Rect({w - borderWidth, 0, borderWidth, h});
418     SDL_FillRect(surf, &borderRect, c);
419
420     tes[0] = SDL_CreateTextureFromSurface(ren, surf);
421
422     SDL_FreeSurface(surf);
423
424 }
```

7.8.3.8 FillWithColor() [2/2]

```
void Sketch::FillWithColor (
             SDL_Color c )
```

fill background color with color C

Parameters

c	SDL_Color, color to fill fill background color with color C

Definition at line 381 of file Sketch.cpp.

```
382 {
383     SDL_Color temp = color;
384     color = c;
385     FillWithColor();
386     color = temp;
387 }
```

7.8.3.9 getCoor()

```
SDL_Rect Sketch::getCoor ( )
```

get coordinate this function will return coordinate of background of sketch

Returns

SDL_Rect

Definition at line 777 of file Sketch.cpp.

```
778 {
779     return coor[0];
780 }
```

7.8.3.10 getText()

```
const std::string & Sketch::getText ( )
```

return text

Definition at line 148 of file Sketch.cpp.

```
149 {
150     return text;
151 }
```

7.8.3.11 hide()

```
void Sketch::hide ( )
```

hide the sketch this function will set visible to false, that will disable the sketch to be rendered

Definition at line 802 of file Sketch.cpp.

```
803 {
804     visible = false;
805 }
```

7.8.3.12 highlight()

```
void Sketch::highlight ( )
```

hightlight the sketch this function will change color of background to invert color

Definition at line 867 of file Sketch.cpp.

```
868 {
869     color.r = 255 - color.r;
870     color.g = 255 - color.g;
871     color.b = 255 - color.g;
872     if(color.r > 20 && color.g > 20 && color.b > 20)
873     {
874         color.r -= color.r * 0.3;
875         color.g -= color.g * 0.3;
876         color.b -= color.b * 0.3;
877     }
878     FillWithColor();
879 }
```

7.8.3.13 init()

```
void Sketch::init (
            const json & mem )
```

init sketch from json

this function call initRect, initColor, initFont, initBorder this function also change visible, if mem contain "visible" key this function will fill with color, if mem contain "fill with color" key

Parameters

mem	json

example of param mem:

{

```
"rect":
{
},
"color":
{
},
"font":
{
},
"border":
{
```

```
 },

 "text": "text",

 "visible": true,

"fill with color": true

}
```

Definition at line 738 of file Sketch.cpp.

```
739 {
740
741     initRect(mem);
742     initColor(mem);
743     initFont(mem);
744     initBorder(mem);
745
746     if(mem.contains("text"))
747     {
748         setText(mem["text"].get<std::string>());
749     }
750     if(mem.contains("visible"))
751         visible = mem["visible"];
752     if(mem.contains("fill with color"))
753     {
754         FillWithColor();
755     }
756 }
```

7.8.3.14 initBorder()

```
void Sketch::initBorder (
             const json & mem )  [protected]
```

init border from json

if mem is not contain "border" key, do nothing

if in "border" object contain "width" key, set width of border to be mem["border"]["width"]

if in "border" object contain "color" key, set color of border to be mem["border"]["color"]

example of param mem:

{

"border": {

```
"width": 0,

"color": {

  "r": 0,

  "g": 0,

  "b": 0,

  "a": 0

}
```

}

}

Parameters

mem	json, contain border of sketch

Definition at line 667 of file Sketch.cpp.

```
668 {
669     if(!mem.contains("border")) return ;
670     if(mem["border"].contains("width"))
671         borderWidth = mem["border"]["width"];
672 
673     if(mem["border"].contains("color"))
674     {
675         if(mem["border"]["color"].contains("r"))
676         {
677             borderColor.r = mem["border"]["color"]["r"];
678         }
679         if(mem["border"]["color"].contains("g"))
680         {
681             borderColor.g = mem["border"]["color"]["g"];
682         }
683         if(mem["border"]["color"].contains("b"))
684         {
685             borderColor.b = mem["border"]["color"]["b"];
686         }
687         if(mem["border"]["color"].contains("a"))
688         {
689             borderColor.a = mem["border"]["color"]["a"];
690         }
691     }
692 }
```

7.8.3.15 initColor()

```
void Sketch::initColor (
            const json & mem )  [protected]
```

init color from json

if mem is not contain "color" key, do nothing

if in "color" object contain "r" key, set r color of sketch to be mem["color"]["r"]

if in "color" object contain "g" key, set g color of sketch to be mem["color"]["g"]

if in "color" object contain "b" key, set b color of sketch to be mem["color"]["b"]

if in "color" object contain "a" key, set a color of sketch to be mem["color"]["a"]

example of param mem:

{

"color": {

```
 "r": 0,
 "g": 0,
 "b": 0,
 "a": 0
```

}

}

Parameters

mem	json, contain color of sketch

Definition at line 512 of file Sketch.cpp.

```
513 {
514     if(mem.contains("color"))
515     {
516         if(mem["color"].contains("r"))
517             color.r = mem["color"]["r"];
518         if(mem["color"].contains("g"))
519             color.g = mem["color"]["g"];
520         if(mem["color"].contains("b"))
521             color.b = mem["color"]["b"];
522         if(mem["color"].contains("a"))
523             color.a = mem["color"]["a"];
524         cache = color;
525     }
526 }
```

7.8.3.16 initFont()

```
void Sketch::initFont (
            const json & mem )  [protected]
```

init font from json

if mem is not contain "font" key, do nothing

get font file and combine with GLOBAL::FontsFolder to get full path of font file

source font from that path and source the size of the font

if in "font" object contain "rect" key, get rect text

if in "font" object contain "color" key, get color text

if int "font" object contain "text", set default text of sketch to be mem["font"]["text"]

example of param mem:

{

"font": {

```
 "name": "font.ttf",
 "size": 0,
 "rect": {
      "x": 0,
      "y": 0,
      "w": 0,
      "h": 0
},
"color": {
```

```
      "r": 0,

      "g": 0,

      "b": 0,

      "a": 0

},

"text": "text"


}

}
```

Parameters

mem	json, contain font of sketch

Definition at line 584 of file Sketch.cpp.

```
585 {
586     if(!mem.contains("font")) return ;
587     if(mem["font"].contains("name") && mem["font"].contains("size"))
588     {
589         char* name = combineLink(GLOBAL::FontsFolder, mem["font"]["name"].get<std::string>().c_str());
590         if(font != nullptr)
591         {
592             TTF_CloseFont(font);
593             font = nullptr;
594         }
595         font = TTF_OpenFont(name, mem["font"]["size"]);
596     }
597     if(mem["font"].contains("rect"))
598     {
599         if(mem["font"]["rect"].contains("x"))
600             coor[1].x = mem["font"]["rect"]["x"];
601         if(mem["font"]["rect"].contains("y"))
602             coor[1].y = mem["font"]["rect"]["y"];
603         if(mem["font"]["rect"].contains("align X"))
604             textAlignX = mem["font"]["rect"]["align X"];
605         if(mem["font"]["rect"].contains("align Y"))
606             textAlignX = mem["font"]["rect"]["align Y"];
607     }
608     if(mem["font"].contains("color"))
609     {
610         if(mem["font"]["color"].contains("r"))
611         {
612             fontColor.r = mem["font"]["color"]["r"];
613         }
614         if(mem["font"]["color"].contains("g"))
615         {
616             fontColor.g = mem["font"]["color"]["g"];
617         }
618         if(mem["font"]["color"].contains("b"))
619         {
620             fontColor.b = mem["font"]["color"]["b"];
621         }
622         if(mem["font"]["color"].contains("a"))
623         {
624             fontColor.a = mem["font"]["color"]["a"];
625         }
626     }
627     if(mem["font"].contains("text"))
628     {
629         setText(mem["font"]["text"].get<std::string>());
630     }
631 }
```

7.8.3.17 initRect()

```
void Sketch::initRect (
```

```
            const json & mem )  [protected]
```

set cooridnate of sketch from json

if mem is not contain "rect" key, do nothing

if in "rect" object contain "x" key, set x coordinate of sketch to be mem["rect"]["x"]

if in "rect" object contain "y" key, set y coordinate of sketch to be mem["rect"]["y"]

if in "rect" object contain "w" key, set w coordinate of sketch to be mem["rect"]["w"]

if in "rect" object contain "h" key, set h coordinate of sketch to be mem["rect"]["h"]

example of param mem:

{

```
"rect": {
     "x": 0,
     "y": 0,
     "w": 0,
     "h": 0
}
```

Parameters

mem	json, contain coordinate of sketch

Definition at line 457 of file Sketch.cpp.

```
458 {
459     if(mem.contains("rect"))
460     {
461         if(mem["rect"].contains("x"))
462         {
463             coor[0].x = mem["rect"]["x"];
464         }
465         if(mem["rect"].contains("y"))
466         {
467             coor[0].y = mem["rect"]["y"];
468         }
469         if(mem["rect"].contains("w"))
470         {
471             coor[0].w = mem["rect"]["w"];
472         }
473         if(mem["rect"].contains("h"))
474         {
475             coor[0].h = mem["rect"]["h"];
476         }
477     }
478 }
```

7.8.3.18 isLieInside()

```
bool Sketch::isLieInside (
            int x,
            int y )
```

determine a point is lie inside sketch or not this function will return true if point (x, y) lie inside sketch

Parameters

x	int
y	int

Returns

bool

Definition at line 813 of file Sketch.cpp.

```
814 {
815     if(x < coor[0].x || coor[0].x + coor[0].w <= x) return false;
816     if(y < coor[0].y || coor[0].y + coor[0].y <= y) return false;
817     return true;
818 }
```

7.8.3.19 isVisible()

```
bool Sketch::isVisible ( )
```

get visible this function will return visible of sketch

Returns

bool

Definition at line 786 of file Sketch.cpp.

```
787 {
788     return visible;
789 }
```

7.8.3.20 moveTo()

```
void Sketch::moveTo (
            int x,
            int y,
            double time )
```

animation of sketch to move the sketch to point (x, y) in time (second) this function will move the sketch to point (x, y) in time (second)

Parameters

x	int
y	int
time	double

Definition at line 826 of file Sketch.cpp.

```
827 {
828     int dx = x - getCoor().x;
829     int dy = y - getCoor().y;
830 
831     if(diff(time, 0))
832     {
833         setX(x);
834         setY(y);
835         return ;
836     }
837 
838     double velo;
839 
840     if(abs(dx) < abs(dy))
841         velo = dy / time;
842     else velo = dx / time;
843 
844     int loop = std::min(80.0, abs(velo * time));
845 
846     time = time / loop;
847 
848     for(int i = 0; i <= loop; i++)
849     {
850         Uint32 startTime = SDL_GetTicks();
851 
852         addX(-dx * (i - 1) / loop);
853         addX(dx * i / loop);
854         addY(-dy * (i - 1) / loop);
855         addY(dy * i / loop);
856         render();
857         Uint32 deltaTime = SDL_GetTicks() - startTime;
858         startTime = SDL_GetTicks();
859         if(deltaTime <= time * 1000)
860             SDL_Delay(time * 1000 - deltaTime);
861     }
862 }
```

7.8.3.21 popChar()

```
void Sketch::popChar ( )
```

erase a character if text is empty then do nothing pop a character from the end of the text after that new text texture will be create

Definition at line 112 of file Sketch.cpp.

```
113 {
114     if(text.empty()) return ;
115     text.pop_back();
116     createTextTexture();
117 }
```

7.8.3.22 render()

```
void Sketch::render ( )
```

render if sketch is hided, do nothing render sketch if renderer is nullptr, it may cause error, require setRender() before render() render background fisrt then render text

Definition at line 328 of file Sketch.cpp.

```
329 {
330     if(!isVisible()) return ;
331     if(tes[0] != nullptr) SDL_RenderCopy(ren, tes[0], nullptr, &coor[0]);
332     if(tes[1] != nullptr) SDL_RenderCopy(ren, tes[1], &crop, &coor[1]);
333 }
```

7.8.3.23 setBorder()

```
void Sketch::setBorder (
            int w,
            int r,
            int g,
            int b,
            int a )
```

set border

set border of sketch

Parameters

w	interger, width of border
r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255
a	interger, alpha value of border color, 0 - 255

Definition at line 355 of file Sketch.cpp.

```
356 {
357     borderWidth = w;
358     borderColor.r = r;
359     borderColor.g = g;
360     borderColor.b = b;
361     borderColor.a = a;
362 }
```

7.8.3.24 setBorderColor()

```
void Sketch::setBorderColor (
            int r,
            int g,
            int b )
```

set border color

Parameters

r	interger, red value of border color, 0 - 255
g	interger, green value of border color, 0 - 255
b	interger, blue value of border color, 0 - 255 set border color

Definition at line 370 of file Sketch.cpp.

```
371 {
372     borderColor.r = r;
373     borderColor.g = g;
374     borderColor.b = b;
375 }
```

7.8.3.25 setColor() [1/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b )
```

set background color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set background color to be (r, g, b)

Definition at line 167 of file Sketch.cpp.

```
168 {
169     color.r = r;
170     color.g = g;
171     color.b = b;
172 }
```

7.8.3.26 setColor() [2/3]

```
void Sketch::setColor (
            int r,
            int g,
            int b,
            int a )
```

set background color to be (r, g, b a)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255
a	interger, alpha value, 0 - 255 set background color to be (r, g, b, a)

Definition at line 182 of file Sketch.cpp.

```
183 {
184     color.r = r;
185     color.g = g;
186     color.b = b;
187     color.a = a;
188 }
```

7.8.3.27 setColor() [3/3]

```
void Sketch::setColor (
            SDL_Color c )
```

set background color to be c

Parameters

c	SDL_Color, background color

Definition at line 156 of file Sketch.cpp.

```
157 {
158     color = c;
159 }
```

7.8.3.28 setCoor()

```
void Sketch::setCoor (
              int x,
              int y,
              int w,
              int h )
```

set coordinate of sketch

Parameters

x	interger, x coordinate of the top left corner of the sketch
y	interger, y coordinate of the top left corner of the sketch
w	interger, width of the sketch
h	interger, height of the sketch set coordinate of sketch

Definition at line 198 of file Sketch.cpp.

```
199 {
200     coor[0] = SDL_Rect({x, y, w, h});
201     align();
202 }
```

7.8.3.29 setH()

```
void Sketch::setH (
              int h )
```

set coordinate of sketch

Parameters

h	interger, height of the sketch set height of sketch

Definition at line 260 of file Sketch.cpp.

```
261 {
262     coor[0].h = h;
263     align();
264 }
```

7.8.3.30 setInCenterX()

```
void Sketch::setInCenterX ( )
```

align text texture align x coordinate of text texture to be in the center of the background texture

Definition at line 269 of file Sketch.cpp.

```
270 {
271     int x = coor[0].x;
272     int w = coor[0].w;
273     coor[1].x = x + (w - coor[1].w) / 2;
274 }
```

7.8.3.31 setInCenterY()

```
void Sketch::setInCenterY ( )
```

align text texture align y coordinate of text texture to be in the center of the background texture

Definition at line 279 of file Sketch.cpp.

```
280 {
281     int y = coor[0].y;
282     int h = coor[0].h;
283     coor[1].y = y + (h - coor[1].h) / 2;
284 }
```

7.8.3.32 setOnLeftSideX()

```
void Sketch::setOnLeftSideX ( )
```

align text texture align x coordinate of text texture to be in the left side of the background texture

Definition at line 289 of file Sketch.cpp.

```
290 {
291     coor[1].x = coor[0].x;
292 }
```

7.8.3.33 setOnLeftSideY()

```
void Sketch::setOnLeftSideY ( )
```

align text texture align y coordinate of text texture to be in the top side of the background texture

Definition at line 307 of file Sketch.cpp.

```
308 {
309     coor[1].y = coor[0].y;
310 }
```

7.8.3.34 setOnRightSideX()

```
void Sketch::setOnRightSideX ( )
```

align text texture align x coordinate of text texture to be in the right side of the background texture

Definition at line 297 of file Sketch.cpp.

```
298 {
299     int x = coor[0].x;
300     int w = coor[0].w;
301     coor[1].x = x + w - coor[1].w;
302 }
```

7.8.3.35 setOnRightSideY()

```
void Sketch::setOnRightSideY ( )
```

align text texture align y coordinate of text texture to be in the bottom side of the background texture

Definition at line 315 of file Sketch.cpp.

```
316 {
317     int y = coor[0].y;
318     int h = coor[0].h;
319     coor[1].y = y + h - coor[1].h;
320 }
```

7.8.3.36 setRender()

```
void Sketch::setRender (
            SDL_Renderer *& r )
```

set render

Parameters

r	address of SDL_Renderer pointer set render of sketch

Definition at line 339 of file Sketch.cpp.

```
340 {
341     ren = r;
342 }
```

7.8.3.37 setText()

```
void Sketch::setText (
            std::string s )
```

set text to be s

Parameters

s	string that will be set to text set text to be s and create new text texture

Definition at line 124 of file Sketch.cpp.

```
125 {
126     text = s;
127     createTextTexture();
128 }
```

7.8.3.38 setTextColor()

```
void Sketch::setTextColor (
            int r,
            int g,
            int b )
```

set text color to be (r, g, b)

Parameters

r	interger, red value, 0 - 255
b	interger, blue value, 0 - 255
g	interger, green value, 0 - 255 set text color to be (r, g, b) and create new text texture

Definition at line 136 of file Sketch.cpp.

```
137 {
138     fontColor.r = r;
139     fontColor.g = g;
140     fontColor.b = b;
141     createTextTexture();
142 }
```

7.8.3.39 setW()

```
void Sketch::setW (
            int w )
```

set coordinate of sketch

Parameters

w	interger, width of the sketch set width of sketch

Definition at line 250 of file Sketch.cpp.

```
251 {
252     coor[0].w = w;
253     align();
254 }
```

7.8.3.40 setX()

```
void Sketch::setX (
            int x )
```

set coordinate of sketch

Parameters

x	interger, x coordinate of the top left corner of the sketch set x coordinate of sketch

Definition at line 208 of file Sketch.cpp.

```
209 {
210     coor[0].x = x;
211     align();
212 }
```

7.8.3.41 setY()

```
void Sketch::setY (
            int y )
```

set coordinate of sketch

Parameters

y	interger, y coordinate of the top left corner of the sketch set y coordinate of sketch

Definition at line 240 of file Sketch.cpp.

```
241 {
242     coor[0].y = y;
243     align();
244 }
```

7.8.3.42 show()

```
void Sketch::show ( )
```

show the sketch this function will set visible to true, that will enable the sketch to be rendered

Definition at line 794 of file Sketch.cpp.

```
795 {
796     visible = true;
797 }
```

7.8.3.43 unHighlight()

```
void Sketch::unHighlight ( )
```

unhightlight the sketch this function will change color of background to normal color

Definition at line 884 of file Sketch.cpp.

```
885 {
886     color = cache;
887     FillWithColor();
888 }
```

The documentation for this class was generated from the following files:

- include/Sketch.hpp
- src/Sketch.cpp

7.9 vector< T > Class Template Reference

a vector class

```
#include <vector.hpp>
```

Public Types

- using value_type = T
- using reference = T &
- using const_reference = const T &
- using iterator = T *
- using const_iterator = const T *
- using size_type = size_t

Public Member Functions

- vector ()
- vector (size_type n)
- vector (const vector &v)=delete
- vector (vector &&v)
- ~vector ()
- vector & operator= (const vector &v)
- iterator begin ()
- const_iterator begin () const
- iterator end ()
- const_iterator end () const
- const_iterator cbegin () const
- const_iterator cend () const
- size_type size () const
- void resize (size_type n)
- void resize (size_type n, T value)
- size_type capacity () const
- bool empty () const
- void reserve (size_type n)

- reference operator[ ] (size_type n)
- const_reference operator[ ] (size_type n) const
- reference front ()
- const_reference front () const
- reference back ()
- const_reference back () const
- void push_back (const value_type &t)
- void push_back (value_type &&val)
- void pop_back ()
- iterator erase (const_iterator position)
- void clear ()

7.9.1 Detailed Description

template<typename T>
class vector< T >

a vector class

Definition at line 9 of file vector.hpp.

7.9.2 Member Typedef Documentation

7.9.2.1 const_iterator

```
template<typename T >
using vector< T >::const_iterator = const T *
```

Definition at line 19 of file vector.hpp.

7.9.2.2 const_reference

```
template<typename T >
using vector< T >::const_reference = const T &
```

Definition at line 17 of file vector.hpp.

7.9.2.3 iterator

```
template<typename T >
using vector< T >::iterator = T *
```

Definition at line 18 of file vector.hpp.

7.9.2.4 reference

```
template<typename T >
using vector< T >::reference = T &
```

Definition at line 16 of file vector.hpp.

7.9.2.5 size_type

```
template<typename T >
using vector< T >::size_type = size_t
```

Definition at line 20 of file vector.hpp.

7.9.2.6 value_type

```
template<typename T >
using vector< T >::value_type = T
```

Definition at line 15 of file vector.hpp.

7.9.3 Constructor & Destructor Documentation

7.9.3.1 vector() [1/4]

```
template<typename T >
vector< T >::vector ( )  [inline]
```

Definition at line 25 of file vector.hpp.

```
25 :  m_size(0U), m_capacity(0U), arr(nullptr) {}
```

7.9.3.2 vector() [2/4]

```
template<typename T >
vector< T >::vector (
            size_type n )  [inline], [explicit]
```

Definition at line 26 of file vector.hpp.

```
26 :  m_size(n), m_capacity(n), arr(n ?  new T[n] :  nullptr) {}
```

7.9.3.3 vector() [3/4]

```
template<typename T >
vector< T >::vector (
            const vector< T > & v )  [delete]
```

7.9.3.4 vector() [4/4]

```
template<typename T >
vector< T >::vector (
            vector< T > && v )  [inline]
```

Definition at line 28 of file vector.hpp.

```
28 :  m_size(v.m_size), m_capacity(v.m_capacity), arr(v.arr) { v.arr = nullptr; }
```

7.9.3.5 ~vector()

```
template<typename T >
vector< T >::~vector ( )  [inline]
```

Definition at line 30 of file vector.hpp.

```
30 { delete[]arr; }
```

7.9.4 Member Function Documentation

7.9.4.1 back() [1/2]

```
template<typename T >
reference vector< T >::back ( )  [inline]
```

Definition at line 55 of file vector.hpp.

```
55 { return arr[m_size - 1U]; }
```

7.9.4.2 back() [2/2]

```
template<typename T >
const_reference vector< T >::back ( ) const  [inline]
```

Definition at line 56 of file vector.hpp.

```
56 { return arr[m_size - 1U]; }
```

7.9.4.3 begin() [1/2]

```
template<typename T >
iterator vector< T >::begin ( )  [inline]
```

Definition at line 35 of file vector.hpp.

```
35 { return arr; }
```

7.9.4.4 begin() [2/2]

```
template<typename T >
const_iterator vector< T >::begin ( ) const  [inline]
```

Definition at line 36 of file vector.hpp.

```
36 { return arr; }
```

7.9.4.5 capacity()

```
template<typename T >
size_type vector< T >::capacity ( ) const  [inline]
```

Definition at line 46 of file vector.hpp.

```
46 { return m_capacity; }
```

7.9.4.6 cbegin()

```
template<typename T >
const_iterator vector< T >::cbegin ( ) const  [inline]
```

Definition at line 39 of file vector.hpp.

```
39 { return arr; }
```

7.9.4.7 cend()

```
template<typename T >
const_iterator vector< T >::cend ( ) const  [inline]
```

Definition at line 40 of file vector.hpp.

```
40 { return arr + m_size; }
```

7.9.4.8 clear()

```
template<typename T >
void vector< T >::clear  [inline]
```

Definition at line 185 of file vector.hpp.

```
186 {
187     for (auto &i :  (*this))
188         i.~T();
189     m_size = 0U;
190 }
```

7.9.4.9 empty()

```
template<typename T >
bool vector< T >::empty ( ) const  [inline]
```

Definition at line 47 of file vector.hpp.

```
47 { return !m_size; }
```

7.9.4.10 end() [1/2]

```
template<typename T >
iterator vector< T >::end ( )  [inline]
```

Definition at line 37 of file vector.hpp.

```
37 { return arr + m_size; }
```

7.9.4.11 end() [2/2]

```
template<typename T >
const_iterator vector< T >::end ( ) const  [inline]
```

Definition at line 38 of file vector.hpp.

```
38 { return arr + m_size; }
```

7.9.4.12 erase()

```
template<typename T >
vector< T >::iterator vector< T >::erase (
            const_iterator position )  [inline]
```

Definition at line 167 of file vector.hpp.

```
168 {
169     position->~T();
170 
171     //pointer arithmetic
172     for (auto i = const_cast<iterator>(position); i < end() - 1; ++i)
173         (*i) = std::move(*(i + 1));
174 
175     //second option
176     /*for (size_t i = position - begin(); i < m_size; ++i)
177 arr[i] = std::move(arr[i + 1]);*/
178 
179     --m_size;
180 
181     return const_cast<iterator>(position);
182 }
```

7.9.4.13 front() [1/2]

```
template<typename T >
reference vector< T >::front ( )  [inline]
```

Definition at line 53 of file vector.hpp.

```
53 { return arr[0U]; }
```

7.9.4.14 front() [2/2]

```
template<typename T >
const_reference vector< T >::front ( ) const  [inline]
```

Definition at line 54 of file vector.hpp.

```
54 { return arr[0U]; }
```

7.9.4.15 operator=()

```
template<typename T >
vector< T > & vector< T >::operator= (
            const vector< T > & v )  [inline]
```

Definition at line 86 of file vector.hpp.

```
87 {
88     if (this != &v) {
89         ~vector();
90         m_size = v.m_size;
91         m_capacity = v.m_capacity;
92         arr = new T[m_capacity];
93         for (size_t i = 0U; i < m_size; ++i)
94             arr[i] = v.arr[i];
95     }
96     return *this;
97 }
```

7.9.4.16 operator[]() [1/2]

```
template<typename T >
reference vector< T >::operator[] (
            size_type n )  [inline]
```

Definition at line 51 of file vector.hpp.

```
51 { return arr[n]; }
```

7.9.4.17 operator[]() [2/2]

```
template<typename T >
const_reference vector< T >::operator[] (
            size_type n ) const  [inline]
```

Definition at line 52 of file vector.hpp.

```
52 { return arr[n]; }
```

7.9.4.18 pop_back()

```
template<typename T >
void vector< T >::pop_back ( )  [inline]
```

Definition at line 61 of file vector.hpp.

```
61 { arr[--m_size].~T(); }
```

7.9.4.19 push_back() [1/2]

```
template<typename T >
void vector< T >::push_back (
            const value_type & t )  [inline]
```

Definition at line 153 of file vector.hpp.

```
154 {
155     enough_capacity();
156     arr[m_size++] = t;
157 }
```

7.9.4.20 push_back() [2/2]

```
template<typename T >
void vector< T >::push_back (
            value_type && val )  [inline]
```

Definition at line 160 of file vector.hpp.

```
161 {
162     enough_capacity();
163     arr[m_size++] = std::move(val);
164 }
```

7.9.4.21 reserve()

```
template<typename T >
void vector< T >::reserve (
            size_type n )  [inline]
```

Definition at line 140 of file vector.hpp.

```
141 {
142     if (n > m_capacity) {
143         T *newbuff = new T[n];
144         for (size_t i = 0U; i < m_size; ++i)
145             newbuff[i] = std::move(arr[i]);
146         delete[]arr;
147         arr = newbuff;
148         m_capacity = n;
149     }
150 }
```

7.9.4.22 resize() [1/2]

```
template<typename T >
void vector< T >::resize (
            size_type n )  [inline]
```

Definition at line 100 of file vector.hpp.

```
101 {
102     if (n < m_size) {
103         while (n < m_size)
104             pop_back();
105         return;
106     }
107     if (n > m_capacity)
108             reserve(n);
109     m_size = n;
110 
111     /*while (n > m_size) //initialize?
112 {
113 arr[m_size++] = {};
114 }*/
115 
116 }
```

7.9.4.23 resize() [2/2]

```
template<typename T >
void vector< T >::resize (
            size_type n,
            T value )  [inline]
```

Definition at line 118 of file vector.hpp.

```
119 {
120     if (n < m_size) {
121         while (n < m_size)
122             pop_back();
123         return;
124     }
125     if (n > m_capacity)
126             reserve(n);
127     m_size = n;
128 
129     for(int i = 0; i < n; i++)
130         arr[i] = value;
131 
132     /*while (n > m_size) //initialize?
133 {
134 arr[m_size++] = {};
135 }*/
136 
137 }
```

7.9.4.24 size()

```
template<typename T >
size_type vector< T >::size ( ) const  [inline]
```

Definition at line 43 of file vector.hpp.

```
43 { return m_size; }
```

The documentation for this class was generated from the following file:

- include/vector.hpp

Chapter 8

File Documentation

8.1 include/Button.hpp File Reference

```
#include <Object.hpp>
#include <SYSTEM.hpp>
#include <SDL2/SDL.h>
#include <SDL_render.h>
#include <SDL2/SDL_image.h>
#include <nlohmann/json.hpp>
```

Classes

- class Button

  class that represents a button. Button is just a object that when is triggered, it will do something.

Typedefs

- using json = nlohmann::json

8.1.1 Typedef Documentation

8.1.1.1 json

```
using json = nlohmann::json
```

Definition at line 13 of file Button.hpp.

8.2 include/Data_Structures.hpp File Reference

```
#include <SDL2/SDL.h>
#include <SDL_render.h>
#include <Display.hpp>
#include <Object.hpp>
#include <SYSTEM.hpp>
#include <Sketch.hpp>
#include <Script.hpp>
```

Classes

- class Data_Structures

  class that handle data structures.

8.3 include/Display.hpp File Reference

```
#include <SDL2/SDL.h>
#include <SDL_surface.h>
#include <SDL_render.h>
#include <SDL2/SDL_image.h>
#include <SYSTEM.hpp>
#include <Button.hpp>
#include <Object.hpp>
```

Classes

- class Display

  class that represents a screen. Screen just a rectangle with some buttons on it. window can have many screens

8.4 include/DuckWin.hpp File Reference

```
#include <SDL_render.h>
#include <SDL2/SDL.h>
#include <SDL2/SDL_quit.h>
#include <SDL2/SDL_video.h>
#include <SDL2/SDL_surface.h>
#include <SDL2/SDL_events.h>
#include <SDL2/SDL_image.h>
#include <SDL2/SDL_ttf.h>
#include <Display.hpp>
#include <Data_Structures.hpp>
#include <SYSTEM.hpp>
#include <InputBox.hpp>
```

Classes

- class MyWindow

  class that handle screen box, input box, data_structures box Finite state machine

8.5 include/InputBox.hpp File Reference

```
#include <Sketch.hpp>
#include <SYSTEM.hpp>
#include <Button.hpp>
#include <SDL2/SDL.h>
```

Classes

- class InputBox

  class that create an input box and render it to the screen Popup a box that can typing in.

8.6 include/Object.hpp File Reference

```
#include <Sketch.hpp>
#include <SDL2/SDL.h>
#include <SDL2/SDL_render.h>
#include <SYSTEM.hpp>
#include <SDL2/SDL_image.h>
```

Classes

- class Object

  class that create an object and render it to the screen texture can be load from image or create new one with text and background color

8.7 include/Script.hpp File Reference

```
#include <SYSTEM.hpp>
#include <Sketch.hpp>
#include <SDL2/SDL.h>
#include <Object.hpp>
```

Classes

- class Script

  class that load an text image and render it to the screen Support highlight lines

8.8 include/Sketch.hpp File Reference

```
#include <SDL2/SDL.h>
#include <SDL2/SDL_ttf.h>
#include <SDL2/SDL_surface.h>
#include <SDL2/SDL_render.h>
#include <SYSTEM.hpp>
```

Classes

- class Sketch

  class that create an text box and render it to the screen

8.9 include/SYSTEM.hpp File Reference

```
#include <iostream>
#include <fstream>
#include <cstring>
#include <stack>
#include <string>
#include <thread>
#include <mutex>
#include <algorithm>
#include <random>
#include <filesystem>
#include <vector.hpp>
#include <nlohmann/json.hpp>
```

Namespaces

- GLOBAL

  store all global variables
- RANDOM

  store all randomize object

Typedefs

- using json = nlohmann::json

  alternative name for nlohmann::json

Functions

- void RANDOM::init ()
  initialize random number generator
- int RANDOM::getInt (int l, int r)
  get random integer in range [l, r]
- double RANDOM::getDouble (double l, double r)
  get random double in range [l, r]
- char * combineLink (const char *dir, const char *name)
  combine a directory and a file to a full file name
- char * combineName (const char *name, const char *type)
  combine a name and a extension to a full file name
- void readJson (const char *const &link, json &mem)
  opean a json file which path is string s and store it in mem
- void readjson (const char *const &dir, const char *const &name, json &mem)
  opean a json file which path is string dir and file name is name. Store it in mem
- void readJson (std::string s, json &mem)
  opean a json file which path is string s and store it in mem
- int getFirstInt (std::string s)
  get the first interger in a string
- void getColor (std::string s, int &r, int &g, int &b)
  get the first 3 number in a string
- bool diff (double a, double b)
  determine if a double is equal to another double

Variables

- const char * GLOBAL::GraphicsFolder = "asset/graphics/"
  path to the folder that store all graphics
- const char * GLOBAL::BackgroundFolder = "asset/graphics/"
  path to the folder that store all background asset
- const char * GLOBAL::ButtonFolder = "asset/graphics/"
  path to the folder that store all button asset
- const char * GLOBAL::AttributeFolder = "asset/attribute/"
  path to the folder that store all custom attribute of graphics, sound, etc
- const char * GLOBAL::AtrbScreens = "asset/attribute/screens/"
  path to the folder that store all custom attribute of screens
- const char * GLOBAL::AtrbButtons = "asset/attribute/buttons/"
  path to the folder that store all custom attribute of buttons
- const std::string GLOBAL::AtrbDT = "asset/attribute/DataStructures/"
  path to the folder that store all custom attribute of data structures
- const char * GLOBAL::FontsFolder = "asset/fonts"
  path to the folder that store all custom attribute of fonts
- const std::string GLOBAL::AtrbInputBox = "asset/attribute/InputBox/"
  path to the folder that store all custom attribute of input box
- const std::string GLOBAL::AtrbScript = "asset/attribute/script/"
  path to the folder that store all custom attribute of script
- const int GLOBAL::WAITING = 800
  time to wait for the next action
- const std::string GLOBAL::SoundFolder = "asset/sound/"
  path to the folder that store all custom attribute of sound
- std::mt19937 RANDOM::rng
  random number generator

8.9.1 Typedef Documentation

8.9.1.1 json

```
using json = nlohmann::json
```

alternative name for nlohmann::json

Definition at line 20 of file SYSTEM.hpp.

8.9.2 Function Documentation

8.9.2.1 combineLink()

```
char* combineLink (
            const char * dir,
            const char * name )
```

combine a directory and a file to a full file name

exam 1: combine "asd" and "bcs" in to "asd/bcs"

exam 2: combine "asd/" and "bsc" into "asd/bsc"

exam 3: combine "asd" and "/bcs" into "asd/bcs"

exam 4: combine "asd/" and "/bcs" into "asd/bcs"

Parameters

dir	cstring, the directory
name	cstring, the file name

Definition at line 135 of file SYSTEM.cpp.

```
136 {
137
138     int n = strlen(dir);
139     int m = strlen(name);
140     char* link = new char[n + m + 5];
141
142     strcpy(link, dir);
143
144     if(dir[n - 1] == '/' && name[0] == '/')
145         link[n - 1] = '\0';
146     else if(dir[n - 1] != '/' && name[0] != '/')
147     {
148         strcat(link, "/");
149     }
150
151     strcat(link, name);
152
153     return link;
```

```
154 }
```

8.9.2.2 combineName()

```
char* combineName (
            const char * name,
            const char * type )
```

combine a name and a extension to a full file name

exam 1: combine "asd" and "bcs" in to "asd.bcs"

exam 2: combine "asd." and "bsc" into "asd.bcs"

exam 3: combine "asd" and ".bcs" into "asd.bcs"

exam 4: combine "asd." and ".bcs" into "asd.bcs"

Parameters

name	cstring, the name
type	cstring, the extension

Definition at line 102 of file SYSTEM.cpp.

```
103 {
104     int n = strlen(name);
105     int m = strlen(type);
106     char* link = new char[n + m + 5];
107
108     strcpy(link, name);
109
110     if(name[n - 1] == '.'  && type[0] == '.')
111         link[n - 1] = '\0';
112     else if(name[n - 1] != '.'  && type[0] != '.')
113     {
114         strcat(link, ".");
115     }
116
117     strcat(link, type);
118
119     return link;
120 }
```

8.9.2.3 diff()

```
bool diff (
            double a,
            double b )
```

determine if a double is equal to another double

Parameters

a	double, the first double
b	double, the second double

Returns

true if a == b (error is less than 1e-6)

Definition at line 204 of file SYSTEM.cpp.

```
205 {
206     return fabs(a - b) < 1e-6;
207 }
```

8.9.2.4 getColor()

```
void getColor (
              std::string s,
              int & r,
              int & g,
              int & b )
```

get the first 3 number in a string

Parameters

s	string, the string to get the first 3 number
r	int&, the first number
g	int&, the second number
b	int&, the third number

Definition at line 244 of file SYSTEM.cpp.

```
245 {
246     int i = 0;
247     r = 0;
248     g = 0;
249     b = 0;
250
251     while(i < (int)s.size() && !isdigit(s[i]))
252         i++;
253     while(i < (int) s.size() && isdigit(s[i]))
254         r = r * 10 + s[i++] - '0';
255
256     while(i < (int)s.size() && !isdigit(s[i]))
257         i++;
258     while(i < (int) s.size() && isdigit(s[i]))
259         g = g * 10 + s[i++] - '0';
260
261     while(i < (int)s.size() && !isdigit(s[i]))
262         i++;
263     while(i < (int) s.size() && isdigit(s[i]))
264         b = b * 10 + s[i++] - '0';
265 }
```

8.9.2.5 getFirstInt()

```
int getFirstInt (
              std::string s )
```

get the first interger in a string

if the string is "asd123", return 123

if the string is "123asd", return 123

if the string is "asd123asd", return 123

if the string is "asd", return 0

if the string is "1a2" return 1

Parameters

s	string, the string to get the first interger

Definition at line 223 of file SYSTEM.cpp.

```
224 {
225     int res = 0;
226     int i = 0;
227
228     while(i < (int) s.size() && !isdigit(s[i]))
229         i++;
230     while(i < (int) s.size() && isdigit(s[i]))
231     {
232         res = res * 10 + s[i] - '0';
233         i++;
234     }
235     return res;
236 }
```

8.9.2.6 readjson()

```
void readjson (
            const char *const & dir,
            const char *const & name,
            json & mem )
```

opean a json file which path is string dir and file name is name. Store it in mem

Parameters

dir	cstring, the path to the directory
name	cstring, the name of the file
mem	json, the json object to store the data

Definition at line 189 of file SYSTEM.cpp.

```
190 {
191     char* link = combineLink(dir, name);
192
193     readJson(link, mem);
194
195     delete [] link;
196 }
```

8.9.2.7 readJson() [1/2]

```
void readJson (
            const char *const & link,
            json & mem )
```

opean a json file which path is string s and store it in mem

Parameters

link	cstring, the path to the json file
mem	json, the json object to store the data

Definition at line 172 of file SYSTEM.cpp.

```
173 {
174     std::ifstream fin(link);
175 
176     fin » mem;
177 
178     fin.close();
179     return ;
180 }
```

8.9.2.8 readJson() [2/2]

```
void readJson (
              std::string s,
              json & mem )
```

opean a json file which path is string s and store it in mem

Parameters

s	string, the path to the json file
mem	json, the json object to store the data

Definition at line 161 of file SYSTEM.cpp.

```
162 {
163     readJson(s.c_str(), mem);
164 }
```

8.10 include/vector.hpp File Reference

```
#include <cstddef>
#include <utility>
```

Classes

- class vector< T >

  a vector class

8.11 README.md File Reference

8.12 src/Button.cpp File Reference

```
#include <Button.hpp>
```

8.13 src/Data_Structures.cpp File Reference

```
#include <Data_Structures.hpp>
```

Functions

- bool isdigit (char ch)

8.13.1 Function Documentation

8.13.1.1 isdigit()

```
bool isdigit (
            char ch )
```

Definition at line 348 of file Data_Structures.cpp.

```
349 {
350     return '0' <= ch && ch <= '9';
351 }
```

8.14 src/Display.cpp File Reference

```
#include <Display.hpp>
```

8.15 src/DuckWin.cpp File Reference

```
#include <DuckWin.hpp>
```

8.16 src/InputBox.cpp File Reference

```
#include <InputBox.hpp>
```

8.17 src/main.cpp File Reference

```
#include <DuckWin.hpp>
```

Functions

- int main ()

8.17.1 Function Documentation

8.17.1.1 main()

```
int main ( )
```

Definition at line 4 of file main.cpp.

```
5 {
6     MyWindow* mainWin = new MyWindow;
7
8     mainWin->init();
9     mainWin->changeScreens("home.json");
10
11     mainWin->run();
12
13     delete mainWin;
14
15     return 0;
16 }
```

8.18 src/Object.cpp File Reference

```
#include <Object.hpp>
```

8.19 src/Script.cpp File Reference

```
#include <Script.hpp>
```

8.20 src/Sketch.cpp File Reference

```
#include <Sketch.hpp>
```

8.21 src/SYSTEM.cpp File Reference

```
#include <SYSTEM.hpp>
```

Functions

- char * combineName (const char *name, const char *type)

  combine a name and a extension to a full file name
- char * combineLink (const char *dir, const char *name)

  combine a directory and a file to a full file name
- void readJson (std::string s, json &mem)

  opean a json file which path is string s and store it in mem
- void readJson (const char *const &link, json &mem)

  opean a json file which path is string s and store it in mem
- void readjson (const char *const &dir, const char *const &name, json &mem)

  opean a json file which path is string dir and file name is name. Store it in mem
- bool diff (double a, double b)

  determine if a double is equal to another double
- int getFirstInt (std::string s)

  get the first interger in a string
- void getColor (std::string s, int &r, int &g, int &b)

  get the first 3 number in a string

8.21.1 Function Documentation

8.21.1.1 combineLink()

```
char* combineLink (
            const char * dir,
            const char * name )
```

combine a directory and a file to a full file name

exam 1: combine "asd" and "bcs" in to "asd/bcs"

exam 2: combine "asd/" and "bsc" into "asd/bsc"

exam 3: combine "asd" and "/bcs" into "asd/bcs"

exam 4: combine "asd/" and "/bcs" into "asd/bcs"

Parameters

dir	cstring, the directory
name	cstring, the file name

Definition at line 135 of file SYSTEM.cpp.

```
136 {
137
138     int n = strlen(dir);
139     int m = strlen(name);
140     char* link = new char[n + m + 5];
141
142     strcpy(link, dir);
```

```
143
144     if(dir[n - 1] == '/' && name[0] == '/')
145         link[n - 1] = '\0';
146     else if(dir[n - 1] != '/' && name[0] != '/')
147     {
148         strcat(link, "/");
149     }
150
151     strcat(link, name);
152
153     return link;
154 }
```

8.21.1.2 combineName()

```
char* combineName (
              const char * name,
              const char * type )
```

combine a name and a extension to a full file name

exam 1: combine "asd" and "bcs" in to "asd.bcs"

exam 2: combine "asd." and "bsc" into "asd.bcs"

exam 3: combine "asd" and ".bcs" into "asd.bcs"

exam 4: combine "asd." and ".bcs" into "asd.bcs"

Parameters

name	cstring, the name
type	cstring, the extension

Definition at line 102 of file SYSTEM.cpp.

```
103 {
104     int n = strlen(name);
105     int m = strlen(type);
106     char* link = new char[n + m + 5];
107
108     strcpy(link, name);
109
110     if(name[n - 1] == '.'  && type[0] == '.')
111         link[n - 1] = '\0';
112     else if(name[n - 1] != '.'  && type[0] != '.')
113     {
114         strcat(link, ".");
115     }
116
117     strcat(link, type);
118
119     return link;
120 }
```

8.21.1.3 diff()

```
bool diff (
              double a,
              double b )
```

determine if a double is equal to another double

Parameters

a	double, the first double
b	double, the second double

Returns

true if a == b (error is less than 1e-6)

Definition at line 204 of file SYSTEM.cpp.

```
205 {
206     return fabs(a - b) < 1e-6;
207 }
```

8.21.1.4 getColor()

```
void getColor (
            std::string s,
            int & r,
            int & g,
            int & b )
```

get the first 3 number in a string

Parameters

s	string, the string to get the first 3 number
r	int&, the first number
g	int&, the second number
b	int&, the third number

Definition at line 244 of file SYSTEM.cpp.

```
245 {
246     int i = 0;
247     r = 0;
248     g = 0;
249     b = 0;
250 
251     while(i < (int)s.size() && !isdigit(s[i]))
252         i++;
253     while(i < (int) s.size() && isdigit(s[i]))
254         r = r * 10 + s[i++] - '0';
255 
256     while(i < (int)s.size() && !isdigit(s[i]))
257         i++;
258     while(i < (int) s.size() && isdigit(s[i]))
259         g = g * 10 + s[i++] - '0';
260 
261     while(i < (int)s.size() && !isdigit(s[i]))
262         i++;
263     while(i < (int) s.size() && isdigit(s[i]))
264         b = b * 10 + s[i++] - '0';
265 }
```

8.21.1.5 getFirstInt()

```
int getFirstInt (
            std::string s )
```

get the first interger in a string

if the string is "asd123", return 123

if the string is "123asd", return 123

if the string is "asd123asd", return 123

if the string is "asd", return 0

if the string is "1a2" return 1

Parameters

s	string, the string to get the first interger

Definition at line 223 of file SYSTEM.cpp.

```
224 {
225     int res = 0;
226     int i = 0;
227
228     while(i < (int) s.size() && !isdigit(s[i]))
229         i++;
230     while(i < (int) s.size() && isdigit(s[i]))
231     {
232         res = res * 10 + s[i] - '0';
233         i++;
234     }
235     return res;
236 }
```

8.21.1.6 readjson()

```
void readjson (
            const char *const & dir,
            const char *const & name,
            json & mem )
```

opean a json file which path is string dir and file name is name. Store it in mem

Parameters

dir	cstring, the path to the directory
name	cstring, the name of the file
mem	json, the json object to store the data

Definition at line 189 of file SYSTEM.cpp.

```
190 {
191     char* link = combineLink(dir, name);
192
193     readJson(link, mem);
```

```
194
195     delete [] link;
196 }
```

8.21.1.7 readJson() [1/2]

```
void readJson (
             const char *const & link,
             json & mem )
```

opean a json file which path is string s and store it in mem

Parameters

link	cstring, the path to the json file
mem	json, the json object to store the data

Definition at line 172 of file SYSTEM.cpp.

```
173 {
174     std::ifstream fin(link);
175
176     fin » mem;
177
178     fin.close();
179     return ;
180 }
```

8.21.1.8 readJson() [2/2]

```
void readJson (
             std::string s,
             json & mem )
```

opean a json file which path is string s and store it in mem

Parameters

s	string, the path to the json file
mem	json, the json object to store the data

Definition at line 161 of file SYSTEM.cpp.

```
162 {
163     readJson(s.c_str(), mem);
164 }
```

Index