Protobyte API

Contributors: Brittni L. Watkins

***API Key:***

* ***Headers***
* **/Directories/and/Paths.ext**
* *Code*
* Menu Options
* Problem/Question

***Variables:***

* Project Title
  + The title of your project
* projectName
  + Name of your project. Will be used to name main project .cpp and .h files

***Beginning a New Project:***

1. Creating Project Files
   1. Open terminal window
   2. Navigate to **project\_wizard** directory of Protobyte
      1. Current Absolute Path
         1. **/Dropbox/ira\_dev/Protobyte\_0.1.0\_quark/Protobyte/project\_wizard/**
   3. Execute command *./wizard projectName*
      1. Creates .cpp and .h files in **/Protobyte\_0.1.0\_quark /Protobyte/src**
2. Adding Project Files to Protobyte Environment
   1. Open Visual Studios
   2. Open project file
      1. Current Absolute Path
         1. **/Dropbox/ira\_dev/Protobyte\_0.1.0\_quark/protobyte\_dev\_WIN/Zhlong\_dev\_WIN/Shlonh\_dev\_WIN.vcxproj**
   3. Add new project files to Visual Studios Project
      1. Right click **srcProtobyte** folder under **Protobyte\_Project**
      2. Hover over Add menu option
      3. Select Existing Item
      4. Navigate to **src** directory
         1. Current Absolute Path
            1. **/Dropbox/ira\_dev/Protobyte\_0.1.0\_quark/Protobyte/src/**
      5. Select and Add projectName.h and projectName.cpp
3. Running new project
   1. Open ProtoMain.cpp
   2. At top of document
      1. *#include “projectName.h”*
   3. In main()
      1. *ijg::ProtoPlasm objectName(canvasWidth, canvasHeight, “Project Title”, new projectName());*
      2. Note: canvasWidth, canvasHeight, and Project Title can only be changed in ProtoMain.cpp

***General Information about Visual Environment:***

* Origin (0, 0) is at the center of the window
* Default background color is black
* Default stoke color is black
* Default fill color is white
* RGB is on a 0-1 scale
* Greyscale
  + 0 = black
  + 1 = white

***Visual Environment:***

Variables:

* Note: the changes made to these variables only apply to those shapes drawn after the changes are made.

bool isStroke

* When set to *true*, shows the shapes’ borders in the current *strokeColor*
* When set to *false*, shapes’ borders do not appear

bool isFill

* When set to *true*, shows the shapes’ fill in the current *fillColor*
* When set to false, shapes’ fill does not appear
  + Shape becomes “see-through”

float lineWidth

* Sets the width of the shapes’ stroke/border
* Example
  + *lineWidth = 2.0;*
  + *lineWidth = 1.0;*

Col4f fillColor

* sets the shapes’ fill color
* Example:
  + *fillColor = Col4f(0.2, 0.5, 0.1, 1.0);*

Col4f strokeColor

* sets the shapes’ stroke/border color
* Example:
  + *strokeColor = Col4f(1.0, 0.0, 0.0, 1.0);*

Functions:

int getWidth();

* returns the width of the window/canvas
* returns same value as *getCanvasWidth()*

int getHeight();

* returns the height of the window/canvas
* returns same value as *getCanvasHeight()*

Dim2i getSize();

* returns a Dim2i object containing the width and height of the window/canvas
* returns same value as *getCanvasSize()* and *getWindowFrameSize()*

int getCanvasWidth();

* returns the width of the window/canvas
* returns same value as *getWidth()*

int getCanvasHeight();

* returns the height of the window/canvas
* returns same value as *getHeight()*

Dim2i getCanvasSize();

* returns a Dim2i object containing the width and height of the window/canvas
* returns same value as *getSize()* and *getWindowFrameSize()*

Dim2i getWindowFrameSize();

* returns a Dim2i object containing the width and height of the window/canvas
* returns same value as *getSize()* and *getCanvasSize()*

void background(float color);

* sets the background of the window
* performs same function as *setBackground(…)*
* Alternative methods of invocation:
  + background(float r, float g, float b);
  + background(Col3f color);
  + background(Col4f color);

void setBackground(float color)

* sets the background of the window
* performs the same function as *background(…)*
* Alternative methods of invocation:
  + setBackground(float r, float g, float b);
  + setBackground(Col3f color);
  + setBackground(Col4f color);

void stroke(float color)

void strokeWeight(float lineWidth)

void noStroke()

void fill(float color)

void noFill()

void setFrameRate(float frameRate);

float getFrameRate();

int getFrameCount();

push()

pop()

translate()

rotate()

scale()

***Objects:***

Dim2i

Col4f

Col4

Col3f

Col3

Vec2f

Vec2

***Mouse/Key Events:***

Global Variables:

float mouseX

float mouseY

float mouseLastFrameX

float mouseLastFrameY

int mouseButton

int mouseAction

int mouseMods

bool isMousePressed

Functions:

void keyPressed()

void mousePressed() {}

void mouseRightPressed()

void mouseReleased()

void mouseRightReleased()

void mouseMoved()

void mouseDragged()

***Two Dimensional Primitives:***

Rectangle

void rect(float radius1, float radius2);

* rect(width, height);
* Draws a rectangle centered at origin with width of *width* and height of *height*

void rect(const Vec2 &pt0, const Vec2 &pt1);

void rect(float x, float y, float w, float h);

***Suggestions:***

General Information about Visual Environment

* Default background color should be different from default stroke color
  + Maybe have default background color be a shade of grey

Visual Environment

* Overload = operator for Col4f = Col3f so that when using global variables to set fill and stroke color, users may pass the variable a Col4f object or a Col3f object

***Problems:***

Visual Environment

* setWindowFrameSize(…) cannot be made private
  + causes error
* setFrameRate(…) has no impact on the frame rate of the program