**CobraSketch**

*By Andrew Sheldon, Ryan Darge, Collin McAloon, and Christopher Neveu*

CobraSketch is a sketching program written in python. The program is designed as a tool for artists to use to sketch pictures on a computer. The program has various features including saving and loading, shade changes, multiple texture brushes and even layers. This program uses OpenGL in the Pyglet Library. The program uses Kytten, also in the Pyglet library, to design the GUI.

The process started months ago when the project was first assigned and groups were given. The first thing we had to find out is what the client wanted. We met up with three art majors to ask what they would like to see in a sketch program. One student requested a simple interface, support for layered sketches, infinite undoing, and selective undo. Another student requested erasing, history, layers, and variable shading. A last student asked for an ability to mimic texture, variable shading, simple interface, and the ability to change the brush size. As a group, we narrowed is down to the following user requirements:

* Basic Requirements
  + Different thickness/shading brushes
  + Saving/Loading
  + Infinitely Resizable Canvas
* History/Undo Features
  + Selective Undo
  + Infinite Undo
* Layering
  + Multiple layers with toggling visibility
* Mimic Textures
* Low Latency
* Aesthetic
  + “Simple” Interface

The next step was to create a feasibility study. This study showed whether the project was going to be possible and the process we were going to take. The developer requirements were also formed. The developer requirements are:

* Employ an intuitive user interface system
* Add and remove layers
* Toggle layer visibility
* Change the thickness/shade/texture of the brush
* Draw with low latency/lag
* Save and load sketch data
* Resize Canvas
* Undo actions selectively

We decided it would be a smart idea to figure out how we were going to program. Each member of the group was to program a quick prototype in a language. Some were able to do this, and some were not. From these prototypes, we chose to use Python as a language using the piglet library for OpenGL and Kytten for the GUI. We also now had a working prototype that could draw on the screen. This was useful later in the design process.

We then had to start creating diagram to outline the interworking of our program. To do this, we first created a use case diagram. This diagram showed all the things the user would be able to do in our program. The next diagram we made was the class diagram. This was when we laid out the objects we were going to use and how they were going to interact. We developed the methods and attributed we would use in the implementation of our project. This was the first time we made major coding decisions. Once the class diagram was completed, we started on a state diagram. This diagram was made to show what state our program was in at all times. Whether it was drawing or erasing or anything else, the diagram would know. A last diagram was a sequence diagram. This diagram shows the sequence of calls and information returns by the various functions in the program.

Once the UML diagrams were completed, it was time to start programming. To make it easier on us, we used Git to easily work together on the project. This allowed us to have source control instead of worrying by using just a cloud like Google Drive or Dropbox. Chris set up the repository since he already had an account with github and the rest of us got accounts and joined. Separate python files were made for each class. These classes were distributed by assigning issues to everyone. Each person was to fill the class they were assigned with the class attributes and methods. When we finished filling the class assigned, we closed the issue and moved on to work on something else. As the smaller classes were finished, they were incorporated into the larger classes.

To start us off a little easier, we went back to our prototype that was coded months before. This gave us a base to code the ability to draw on the screen. This simple action was incorporated into the Sketch class. It was the start of the way our program interpolates on the screen. Once basic interpolation was designed, it was time to start adding in the special features. This was when the layer class was incorporated along with brush. This created that ability to change the brush size and shade of gray. Other features like undo and redo were added using the history and action classes that saved the information. Save and open features were incorporated to give the user that ability to load a picture to edit and save a picture as a .jpg or other such type.

The next step was to create a GUI. The GUI was programmed using Kytten. Kytten creates drop-down menus and buttons for various features. In the upper left hand corner, a drop down menu was creating with categories of “File”, “Edit”, and “Brush”. The “File” menu includes save and open to activate the save as a picture and load picture features of the program. The “Edit” menu includes Cut, Paste, and Copy. The “Brush” menu includes increase brush size and decrease brush size which activate the change brush size feature of the program. In the upper right hand side of the program is the layers control. This controls if the layers are visible or not and what order they fall in. There are also buttons to delete the layers.

Not all of the features that were originally planned to be put in were implemented. There is still the ability to do so. We were able to get the fundamentals of the project done in the time allotted. The software engineering process was executed well in for this project. Taking the steps of collecting user requirements, creating developer requirements and diagraming everything to plan the project saved a lot of time in the coding process. The group was successful to use the Python language with Pyglet and Kytten libraries to create a basic sketching program called CobraSketch.