**Final Graphics Report**

**Major developments/breakthroughs(reference specific code please):**

1. Threading. Seems quite juvenile but back then it was quite a breakthrough. With this, I could multitask as much as I want.

static MovingKeys mK = new MovingKeys();

Thread t1 = new Thread(mK);

t1.start();

public void run(){

try{

while(true){

move();

Thread.sleep(5);

}

}catch(Exception e){System.err.println(e.getMessage());}

}

1. Shooting Bullets. The sheer amount of time struggling with this code is why it is a breakthrough. The important part of the code:

public void keyPressed(KeyEvent e) {

int keyCode = e.getKeyCode();

if (keyCode == e.VK\_SPACE) {

by = MovingKeys.y;

bx = MovingKeys.x + 30;

System.out.println(bx + " " + by);

bullet = new Rectangle(0, by, 3, 5);

shot = true;

}

}

//@Override

public void keyReleased(KeyEvent e) {

int keyCode = e.getKeyCode();

if (keyCode == e.VK\_SPACE) {

if (bullet.x >= 500) {

bullet = new Rectangle(0, 0, 0, 0);

shot = false;

}

}

}

1. Panels. Struggled with turning the code I had to suit multiple panels. Ended up changing the places of lots of code to make it work. Also, ran into various syntax issues. See my journal for specifics.
2. Sidescrolling – Wrote code from scratch that works fairly well. There are some glitches that will be detailed in major challenges below.

public static int nPic = 0, nPic2 = 1480;

public Image img, img2;

public ScrollingBackground() {

ImageIcon i = new ImageIcon("sky.jpg");

img = i.getImage();

}

public void draw(Graphics g) {

Graphics2D g2 = (Graphics2D) g;

g2.drawImage(img, nPic, 0, null);

if (nPic <= -1000){

g2.drawImage(img, nPic2, 0, null);

}

if (nPic2<= -1000){

nPic = 0;

nPic2 = 1480;

}

System.out.println(nPic + " " + nPic2);

}

public void move() {

nPic -=(1 \* MovingKeys.turbo);

nPic2 -=(1 \* MovingKeys.turbo);

}

I basically had 2 images that looped continuously. One following the other and then resetting.

**Major Challenges/setbacks( reference specific code please):**

The challenges that I had that were solved are all detailed in my journal entries. The only unsolved problem is my side scrolling action. The code that I tried to use but was unsuccessful was:

public static int nPic = 0, nPic2 = 1480;

if (nPic <= -1480 ){

nPic = 1480;

}if(nPic2 <= -1480){

nPic2 = 1480;

}

In theory this should work and Gaurab Aryal did something very similar. The animation would be smooth for transtioning from nPic to nPic2 (nice push effect) put not from nPic2 to nPic (nPic would just suddenly appear) creating this glitch, laggy feel. This was somewhat resolved in the code above for successes but it is still glitchy. However, if I only painted 1 picture, the animation would go like this: image is pushed through… blank for where the other image would be pushed through… and then … a nice push back to the original image. Therefore I conclude my struggles are drawing 2 different images.

EDIT: Got this to work. It was the paint images that I thought was the trouble spot.

Switched places of painting the images to get it to work:

public static int nPic = 0, nPic2 = 1480;

public static boolean bs = false;

public Image img, img2;

public ScrollingBackground() {

ImageIcon i = new ImageIcon("sky.jpg");

img = i.getImage();

}

public void draw(Graphics g) {

Graphics2D g2 = (Graphics2D) g;

g2.drawImage(img, nPic, 0, null);

g2.drawImage(img, nPic2, 0, null);

if (bs == true){

g2.drawImage(img, nPic2, 0, null);

g2.drawImage(img, nPic, 0, null);

}

}

**Any modifications to your planned steps:**

I decided there was no need for “cool pictures”, my last step on my procedures as my partner, Peter Nguyen would be dealing with this in his graphics report. Also, I decided to not do a drop down menu as we would not be doing this in our app. Finally, due to time constraints, I was unable to get a homescreen with a button to click play into my game.