Logic Tab

```
import processing.phone.*;
import processing.video.*;
import processing.image2.*;
Phone p;

void setup()
{
  size(240, 320);
  p = new Phone(this);
  p.fullscreen();        // Go full screen

    loadMenuSetup();
  loadMenuCamera();
  loadTakepicture();
  loadLook();

}

int SCREENSETUP = 0;
int SCREENCAMERA = 1;
int SCREENTAKEPICTURE = 200;

int SCREENMENU = 2;
int SCREENPROFILE = 3;
int SCREENPIAZZA = 4;
int SCREENADDGARMENT = 5;
int SCREENHISTORY = 6;
int SCREENADDGARMENT2 = 7;
int SCREENADDGARMENT3 = 8;
```

```
int SCREENPIAZZAUSERS = 9;
int SCREENUSERSINSIDE1 = 10;
int SCREENMARKERANIMATION = 11;
int SCREENLOOK = 12;

int MENU_OPTION_WARDROBE = 0;

int MENU_OPTION_PIAZZA = 1;
int MENU_OPTION_MIRROR = 2;
int MENU_OPTION_TABLE = 3;
int MENU_OPTION_CLOCK = 4;

int PIAZZA_OPTION_ALTUS = 0;
int PIAZZA_OPTION_PAPEUR = 1;
int PIAZZA_OPTION_OPULENS = 2;

int USERS_OPTION_NICOLA = 0;
int USERS_OPTION_FABRIZIO = 1;
int USERS_OPTION_ALBERTO = 2;
int USERS_OPTION_LUIGI = 3;

int USERS_SELECTED_KNIFE = 0;
int USERS_SELECTED_SLAP = 1;
int USERS_SELECTED_EGG = 2;
int USERS_SELECTED_MARKER = 3;
int USERS_SELECTED_EXAMPLE = 4;

int SCREENUSERSVANDALIZE = 0;
int SCREEN_USERS_VANDALIZE_ANI = 1;
```

```
int ADD_GARMENT_ADD = 0;
int ADD_GARMENT_SCAN = 1;
int ADD_GARMENT_CONFIRM = 2;

int WARDROBE_OPTION_HAT = 0;
int WARDROBE_OPTION_SHIRT = 1;
int WARDROBE_OPTION_TIE = 2;
int WARDROBE_OPTION_JACKET = 3;
int WARDROBE_OPTION_PANTS = 4;
int WARDROBE_OPTION_SHOES = 5;

int WARDROBE_HAT_OPTION_0 = 0;
int WARDROBE_HAT_OPTION_1 = 1;
int WARDROBE_HAT_OPTION_2 = 2;

int WARDROBE_SHIRT_OPTION_0 = 0;
int WARDROBE_SHIRT_OPTION_1 = 1;
int WARDROBE_SHIRT_OPTION_2 = 2;

int menu_option_selected = MENU_OPTION_WARDROBE;
int piazza_option_selected = PIAZZA_OPTION_ALTUS;
int users_option_selected = USERS_OPTION_NICOLA;
int users_selected = USERS_SELECTED_KNIFE;
int users_vandalize = SCREENUSERSVANDALIZE;
int garment_option_selected = ADD_GARMENT_ADD;
int wardrobe_option_selected = WARDROBE_OPTION_HAT;
int wardrobe_hat_option_selected = WARDROBE_HAT_OPTION_0;
int wardrobe_shirt_option_selected = WARDROBE_SHIRT_OPTION_0;
```

```
int screenMode = SCREENSETUP;

// This gets called (once) when a key is pressed.

void keyPressed() {
 if (screenMode == SCREENSETUP){
  if (keyCode == FIRE){
   enterMenuCamera();
  }
 }
 else if (screenMode == SCREENCAMERA){
  if (keyCode == SOFTKEY2){
   enterMenuScreen();
  }
  if (keyCode == SOFTKEY1){
   screenMode = SCREENTAKEPICTURE;
  }
 }
 else if (screenMode == SCREENTAKEPICTURE){
  keyPressedCamera();
 }
 else if (screenMode == SCREENPIAZZA){
  if (keyCode == SOFTKEY2){
   enterMenuScreen();
  }
  else if(piazza_option_selected == PIAZZA_OPTION_ALTUS){
   if (keyCode == LEFT){
    piazza_option_selected = PIAZZA_OPTION_OPULENS;
   }
   else if (keyCode == RIGHT){
```

```
    piazza_option_selected = PIAZZA_OPTION_PAPEUR;
   }
   else if (keyCode == SOFTKEY2){
    screenMode = SCREENPIAZZA;
   }
  }
  else if(piazza_option_selected == PIAZZA_OPTION_PAPEUR){
   if (keyCode == LEFT){
    piazza_option_selected = PIAZZA_OPTION_ALTUS;
   }
   else if (keyCode == RIGHT){
    piazza_option_selected = PIAZZA_OPTION_OPULENS;
   }
   else if (keyCode == FIRE){
    screenMode = SCREENPIAZZAUSERS;
   }
   else if (keyCode == SOFTKEY2){
    screenMode = SCREENPIAZZA;
   }
  }
  else if(piazza_option_selected == PIAZZA_OPTION_OPULENS){
   if (keyCode == LEFT){
    piazza_option_selected = PIAZZA_OPTION_PAPEUR;
   }
   else if (keyCode == RIGHT){
    piazza_option_selected = PIAZZA_OPTION_ALTUS;
   }
   else if (keyCode == SOFTKEY2){
    screenMode = SCREENPIAZZA;
   }
```

```
 }
}
if (screenMode == SCREENPIAZZAUSERS){
 if (users_option_selected == USERS_OPTION_NICOLA){
  if (keyCode == RIGHT){
   users_option_selected = USERS_OPTION_FABRIZIO;
  }
  else if (keyCode == DOWN){
   users_option_selected = USERS_OPTION_ALBERTO;
  }
  else if (keyCode == SOFTKEY2){
   screenMode = SCREENPIAZZA;
  }
 }

 else if (users_option_selected == USERS_OPTION_FABRIZIO){
  if (keyCode == LEFT){
   users_option_selected = USERS_OPTION_NICOLA;
  }
  else if (keyCode == DOWN){
   users_option_selected = USERS_OPTION_LUIGI;
  }
  else if (keyCode == FIRE){
   screenMode = SCREENUSERSINSIDE1;
  }
  else if (keyCode == SOFTKEY2){
   screenMode = SCREENPIAZZA;
  }
 }
```

```
  else if (users_option_selected == USERS_OPTION_ALBERTO){
    if (keyCode == UP){
      users_option_selected = USERS_OPTION_NICOLA;
    }
    else if (keyCode == RIGHT){
      users_option_selected = USERS_OPTION_LUIGI;
    }
    else if (keyCode == SOFTKEY2){
      screenMode = SCREENPIAZZA;
    }
  }


  else if (users_option_selected == USERS_OPTION_LUIGI){
    if (keyCode == UP){
      users_option_selected = USERS_OPTION_FABRIZIO;
    }
    else if (keyCode == LEFT){
      users_option_selected = USERS_OPTION_ALBERTO;
    }
    else if (keyCode == SOFTKEY2){
      screenMode = SCREENPIAZZA;
    }
  }
}

if (screenMode == SCREENUSERSINSIDE1){
  if (users_selected == USERS_SELECTED_KNIFE){
    if (keyCode == DOWN){
      users_selected = USERS_SELECTED_SLAP;
```

```
  }
  else if (keyCode == UP){
    users_selected = USERS_SELECTED_MARKER;
  }
  else if (keyCode == SOFTKEY2){
    screenMode = SCREENPIAZZAUSERS;
  }
 }
 else if (users_selected == USERS_SELECTED_SLAP){
  if (keyCode == DOWN){
    users_selected = USERS_SELECTED_EGG;
  }
  else if (keyCode == UP){
    users_selected = USERS_SELECTED_KNIFE;
  }
  else if (keyCode == SOFTKEY2){
    screenMode = SCREENPIAZZAUSERS;
  }
 }
 else if (users_selected == USERS_SELECTED_EGG){
  if (keyCode == DOWN){
    users_selected = USERS_SELECTED_MARKER;
  }
  else if (keyCode == UP){
    users_selected = USERS_SELECTED_SLAP;
  }
  else if (keyCode == SOFTKEY2){
    screenMode = SCREENPIAZZAUSERS;
  }
 }
```

```
  else if (users_selected == USERS_SELECTED_MARKER){
   if (keyCode == DOWN){
    users_selected = USERS_SELECTED_KNIFE;
   }
   else if (keyCode == UP){
    users_selected = USERS_SELECTED_EGG;
   }
   else if (keyCode == SOFTKEY2){
    screenMode = SCREENPIAZZAUSERS;
   }

   else if (keyCode == FIRE){
    screenMode = SCREENMARKERANIMATION;
   }
  }
 }
 if (screenMode == SCREENMARKERANIMATION){
  if (users_vandalize == SCREENUSERSVANDALIZE){
   if (keyCode == FIRE){
    users_vandalize = SCREEN_USERS_VANDALIZE_ANI;
   }
  }
  if (keyCode == SOFTKEY2){
   screenMode = SCREENUSERSINSIDE1;
  }
 }

else if (screenMode == SCREENHISTORY){
  if (keyCode == SOFTKEY2){
   enterMenuScreen();
```

```
  }
}
else if (screenMode == SCREENADDGARMENT){
  if (keyCode == SOFTKEY2){
    enterMenuScreen();
  }

  else if (garment_option_selected == ADD_GARMENT_ADD){
    if (keyCode == FIRE){
      garment_option_selected = ADD_GARMENT_SCAN;
    }
  }
  else if (garment_option_selected == ADD_GARMENT_SCAN){
    if (keyCode == FIRE){
      garment_option_selected = ADD_GARMENT_CONFIRM;
    }
  }
  else if (garment_option_selected == ADD_GARMENT_CONFIRM){
    if (keyCode == FIRE){
      enterMenuScreen();
    }
  }
}

else if (screenMode == SCREENHISTORY){
  if (keyCode == SOFTKEY2){
    enterMenuScreen();
  }
}
else if (screenMode == SCREENPROFILE){
```

```
  if (keyCode == SOFTKEY2){
    enterMenuScreen();
  }
}

else if (screenMode == SCREENMENU){
  if(menu_option_selected == MENU_OPTION_WARDROBE){
    if(keyCode == RIGHT) {
      menu_option_selected = MENU_OPTION_PIAZZA;
    }
  }
  else if(menu_option_selected == MENU_OPTION_PIAZZA){
    if(keyCode == LEFT) {
      menu_option_selected = MENU_OPTION_WARDROBE;
    }
    if(keyCode == RIGHT) {
      menu_option_selected = MENU_OPTION_MIRROR;
    }
    else if (keyCode == FIRE){
      enterPiazzaScreen();
    }
  }
  else if(menu_option_selected == MENU_OPTION_MIRROR){
    if(keyCode == LEFT) {
      menu_option_selected = MENU_OPTION_PIAZZA;
    }
    if(keyCode == RIGHT) {
      menu_option_selected = MENU_OPTION_TABLE;
    }
    else if (keyCode == FIRE){
```

```
      enterProfileScreen();
     }
    }
    else if(menu_option_selected == MENU_OPTION_TABLE){
     if(keyCode == LEFT) {
      menu_option_selected = MENU_OPTION_MIRROR;
     }
     if(keyCode == RIGHT) {
      menu_option_selected = MENU_OPTION_CLOCK;
     }
     if (keyCode == FIRE) {
      enterScanGarmentScreen();
     }
    }
    else if(menu_option_selected == MENU_OPTION_CLOCK){
     if(keyCode == LEFT) {
      menu_option_selected = MENU_OPTION_TABLE;
     }
     else if (keyCode == FIRE){
      enterHistoryScreen();
     }
    }
  }
}

void draw()

{

  if (screenMode == SCREENSETUP){
```

```
  drawSetup();
}

if (screenMode == SCREENCAMERA){
  drawCamera();
}

if(screenMode == SCREENTAKEPICTURE){
  drawTakepicture();
}

if (screenMode == SCREENMENU){
  drawMenu();
}

else if (screenMode == SCREENPROFILE){
  drawProfileScreen();
}

else if (screenMode == SCREENPIAZZA){
  drawPiazzaScreen();
}

else if (screenMode == SCREENADDGARMENT){
  drawScanGarmentScreen();
}

else if (screenMode == SCREENHISTORY){
  drawHistoryScreen();
}
```

```
 else if (screenMode == SCREENPIAZZAUSERS){
  drawPiazzaUsersScreen();
 }


 else if (screenMode == SCREENUSERSINSIDE1){
  drawUsersInsideScreen();
 }

 else if (screenMode == SCREENMARKERANIMATION){
  drawUserVandalizeScreen();
 }

 else if (screenMode == SCREENLOOK){
  drawLook();
 }
}
```

Graphics Tab

```
PImage b;
PImage man;
PImage buttons;
PImage wardrobetitle;
PImage piazzatitle;
PImage profiletitle;
PImage addtitle;
PImage historytitle;
PImage photoload;

int x=-1200;  // menu targets
```

```
int xTarget=0; // menu targets

int xTitle = -240;
int xTitleTarget = 0;

int y=-640;
int yTarget=0;

int menu_option_selected_previous = -1;

void enterloadLook(){
 screenMode = SCREENLOOK;
 xTitle = -240;
}

void loadLook(){
 photoload = loadImage("lookload.png");
}

void drawLook(){
   image(personPhoto, 0, 0, width, height);
   image (photoload,0,0);
}



void loadMenu()
{
 b = loadImage("pan.png");
 man = loadImage("man.png");
 buttons = loadImage("buttons.png");
```

```
  wardrobetitle = loadImage("wardrobetitle.png");
  piazzatitle = loadImage("piazzatitle.png");
  profiletitle = loadImage("profiletitle.png");
  addtitle = loadImage("addtitle.png");
  historytitle = loadImage("historytitle.png");
}

void enterMenuScreen(){
  screenMode = SCREENMENU;
  xTitle = -240;
}

void drawMenu()
{
  if(menu_option_selected != menu_option_selected_previous){
    menu_option_selected_previous = menu_option_selected;
    xTitle = -240;
  }
  if(menu_option_selected == MENU_OPTION_WARDROBE){
    xTarget = 0;
  }
  if(menu_option_selected == MENU_OPTION_PIAZZA){
    xTarget = -240;
  }
  if(menu_option_selected == MENU_OPTION_MIRROR){
    xTarget = -480;
  }
  if(menu_option_selected == MENU_OPTION_TABLE){
    xTarget = -720;
  }
```

```
 if(menu_option_selected == MENU_OPTION_CLOCK){
  xTarget = -960;
 }
 x = easeOut(x, xTarget, 8);
 xTitle = easeOut(xTitle, xTitleTarget, 6);
 image(b, x, 0);
 image(man,0,0);
 image(buttons,0,0);
 if(menu_option_selected == MENU_OPTION_WARDROBE){
  image(wardrobetitle,xTitle,0);
 }
 if(menu_option_selected == MENU_OPTION_PIAZZA){
  image(piazzatitle,xTitle,0);
 }
 if(menu_option_selected == MENU_OPTION_MIRROR){
  image(profiletitle,xTitle,0);
 }
 if(menu_option_selected == MENU_OPTION_TABLE){
  image(addtitle,xTitle,0);
 }
 if(menu_option_selected == MENU_OPTION_CLOCK){
  image(historytitle,xTitle,0);
 }

}

PImage profile;
PImage profilebuttons;

void loadProfileScreen ()
```

```
{
  profile = loadImage("profile.png");
  profilebuttons = loadImage("profilebuttons.png");
}

void enterProfileScreen(){
  screenMode = SCREENPROFILE;
  y = 640;
}

void drawProfileScreen()

{

  y = easeOut(y, yTarget, 8);
  image (profile,0,y);
  y = easeOut(y, yTarget, 8);
  image (profilebuttons, 0, y);
}

PImage piazza;
PImage piazza2;
PImage piazza3;

void loadPiazzaScreen ()
{
  piazza = loadImage("piazza.png");
  piazza2 = loadImage ("piazza2.png");
  piazza3 = loadImage ("piazza3.png");
}
```

```
void enterPiazzaScreen(){
 screenMode = SCREENPIAZZA;
 y = 640;
}

void drawPiazzaScreen()
{

 y = easeOut(y, yTarget, 8);
 if(piazza_option_selected == PIAZZA_OPTION_PAPEUR){
  image (piazza,0,y);
 }
 else if(piazza_option_selected == PIAZZA_OPTION_ALTUS){
  image (piazza2,0,y);
 }
 else if(piazza_option_selected == PIAZZA_OPTION_OPULENS){
  image (piazza3,0,y);
 }
 image(buttons,0,0);
}

PImage piazzausers;
PImage piazzausers2;
PImage piazzausers3;
PImage piazzausers4;

void loadPiazzaUsersScreen()
{
 piazzausers = loadImage ("piazzausers.png");
```

```
  piazzausers2 = loadImage ("piazzausers2.png");
  piazzausers3 = loadImage ("piazzausers3.png");
  piazzausers4 = loadImage ("piazzausers4.png");
}

void drawPiazzaUsersScreen()
{




  y = easeOut(y, yTarget, 8);
  if (users_option_selected == USERS_OPTION_NICOLA){
    image (piazzausers,0,y);
  }
  else if (users_option_selected == USERS_OPTION_FABRIZIO){
    image (piazzausers2,0,y);
  }
  else if (users_option_selected == USERS_OPTION_ALBERTO){
    image (piazzausers3,0,y);
  }
  else if (users_option_selected == USERS_OPTION_LUIGI){
    image (piazzausers4,0,y);
  }
  image(buttons,0,0);
}




PImage usersinside1;
PImage usersinside2;
```

```
PImage usersinside3;
PImage usersinside4;
PImage usersinside5;

void loadUsersInsideScreen()
{
  usersinside1 = loadImage ("usersinside1.png");
  usersinside2 = loadImage ("usersinside2.png");
  usersinside3 = loadImage ("usersinside3.png");
  usersinside4 = loadImage ("usersinside4.png");
  usersinside5 = loadImage ("usersinside5.png");
}

void drawUsersInsideScreen()
{
  y = easeOut(y, yTarget, 8);
  if (users_selected == USERS_SELECTED_KNIFE){
    image (usersinside1,0,y);
  }
  else if (users_selected == USERS_SELECTED_SLAP){
    image (usersinside2,0,y);
  }
  else if (users_selected == USERS_SELECTED_EGG){
    image (usersinside3,0,y);
  }
  else if (users_selected == USERS_SELECTED_MARKER){
    image (usersinside4,0,y);
  }
  else if (users_selected == USERS_SELECTED_EXAMPLE){
    image (usersinside5,0,y);
```

```
  }
  image(buttons,0,0);
}



PImage mfirst;


int numFrames = 15;
int frame = 0;
PImage[] images = new PImage [numFrames];


void loadUserVandalizeScreen (){
  mfirst = loadImage("mfirst.png");


  framerate(30);
  //noLoop();
  images[0] = loadImage("m1.png");
  images[1] = loadImage("m2.png");
  images[2] = loadImage("m3.png");
  images[3] = loadImage("m4.png");
  images[4] = loadImage("m5.png");
  images[5] = loadImage("m6.png");
  images[6] = loadImage("m7.png");
  images[7] = loadImage("m8.png");
  images[8] = loadImage("m9.png");
  images[9] = loadImage("m10.png");
  images[10] = loadImage("m11.png");
  images[11] = loadImage("m12.png");
  images[12] = loadImage("m13.png");
  images[13] = loadImage("m14.png");
```

```
 images[14] = loadImage("m15.png");

}

void drawUserVandalizeScreen(){

 y = easeOut(y, yTarget, 8);
 if (users_vandalize == SCREENUSERSVANDALIZE){
  image (mfirst,0,y);
 }

 if (users_vandalize == SCREEN_USERS_VANDALIZE_ANI){

  frame++;
  if (frame == numFrames){
   frame = 14;

  }
  image(images[frame],0,0);

 }

}

/*PImage usersinside2;

 void loadUsersInside2Screen()
 {
 usersinside2 = loadImage ("usersinside2.png");
 }
```

```
void drawUsersInside2Screen()
{
y = easeOut(y, yTarget, 8);
image (usersinside2,0,y);
image(buttons,0,0);
}

PImage usersinside3;

void loadUsersInside3Screen()
{
usersinside3 = loadImage ("usersinside3.png");
}

void drawUsersInside3Screen()
{
y = easeOut(y, yTarget, 8);
image (usersinside3,0,y);
image(buttons,0,0);
}

PImage usersinside4;

void loadUsersInside4Screen()
{
usersinside4 = loadImage ("usersinside4.png");
}

void drawUsersInside4Screen()
```

```
{
y = easeOut(y, yTarget, 8);
image (usersinside4,0,y);
image(buttons,0,0);
}

PImage usersinside5;

void loadUsersInside5Screen()
{
usersinside5 = loadImage ("usersinside5.png");
}

void drawUsersInside5Screen()
{
y = easeOut(y, yTarget, 8);
image (usersinside5,0,y);
image(buttons,0,0);
}
*/


PImage scanGarment;
PImage garmentconfirm;
PImage addgarment;

void loadScanGarmentScreen (){
 scanGarment = loadImage("scangarment.png");
 garmentconfirm = loadImage("addgarmentconfirm.png");
 addgarment = loadImage("addgarment.png");
```

```
}

void enterScanGarmentScreen(){
 screenMode = SCREENADDGARMENT;
 y = 640;
}

void drawScanGarmentScreen(){

 y = easeOut(y, yTarget, 8);
 if (garment_option_selected == ADD_GARMENT_ADD){
  image (scanGarment,0,y);
 }
 else if (garment_option_selected == ADD_GARMENT_SCAN){
  image (addgarment,0,y);
 }
 else if (garment_option_selected == ADD_GARMENT_CONFIRM){
  image (garmentconfirm,0,y);
 }
}
/* PImage garmentconfirm;

 void loadGarmentConfirmScreen (){
 garmentconfirm = loadImage("addgarmentconfirm.png");
 }

 void drawGarmentConfirmScreen(){

 y = easeOut(y, yTarget, 8);
 image (garmentconfirm,0,y);
```

```
}

PImage addgarment;

void loadAddingScreen (){
addgarment = loadImage("addgarment.png");
}

void drawAddingScreen(){

y = easeOut(y, yTarget, 8);
image (addgarment,0,y);
}

*/

PImage history;

void enterHistoryScreen(){
 screenMode = SCREENHISTORY;
 y = 640;
}

void loadHistoryScreen (){
 history = loadImage("history.png");
}

void drawHistoryScreen(){

 y = easeOut(y, yTarget, 8);
```

```
  image (history,0,y);
  image(buttons,0,0);
}

//carico la schermata iniziale
PImage start;
PImage picture;

void loadMenuSetup()
{
  buttons = loadImage("buttons.png");
  start = loadImage("start.png");
}
//disegna la schermata iniziale
void drawSetup ()
{
  image(start,0,0);
  /*image(buttons,0,0);*/
}
//carico la schermata per la Camera
void loadMenuCamera()
{
  picture = loadImage("picture.png");

}
void enterMenuCamera(){
  screenMode = SCREENCAMERA;
  xTitle = -240;
}
```

```
//disegna la schermata per la Camera
void drawCamera ()
{
 image(picture,0,0);
}
```

Camera Tab

```
int CAPTUREMODE_CAMERA = 0;
int CAPTUREMODE_DISPLAY = 1;
int captureMode = CAPTUREMODE_DISPLAY;

Capture cap;
PImage personPhoto;
PImage silouette;
PImage save_button;

void loadTakepicture()
{
 cap = null;
 silouette = loadImage("silouette.png");
 save_button = loadImage("save_button.png");
}

void destroy() {
 // close the camera object on sketch exit
 cap.close();
}

void drawTakepicture() {
 if(cap == null){
   cap = new Capture(this, "capture://image"); // set up camera capture from library
```

```
    keyPressedCamera();
  }
  else if (captureMode == CAPTUREMODE_DISPLAY) {
    // In display mode, draw the last captured picture.
    image(personPhoto, 0, 0, width, height);
    image (silouette,0,0);
    image(save_button,0,0);
  }
  // In capture mode, don't draw anything, since the phone does it for us
}

void keyPressedCamera() {
  if (captureMode == CAPTUREMODE_CAMERA) {
    // capture a frame and load it into a PImage
    byte[] data = cap.read();
    cap.hide();
    personPhoto = loadImage(data);
    captureMode = CAPTUREMODE_DISPLAY;
    // in display mode, we need to loop through the draw() function
    loop();
  }
  else if (captureMode == CAPTUREMODE_DISPLAY) {
    if(keyCode == FIRE){
enterloadLook();
    }
    else{
      // In display mode, pressing a key goes back to capture mode
      noLoop();
      cap.show(0, 0, width, height);
      image(silouette, 0, 0);
```

```
    captureMode = CAPTUREMODE_CAMERA;
   }
  }
}
```

Image Scale Tab

```
void image(PImage sourceImage, int xPos, int yPos, int newWidth, int newHeight) {
  int oldWidth = sourceImage.width;
  int oldHeight = sourceImage.height;

  PImage2 inputData = new PImage2(sourceImage);
  PImage2 outputData = new PImage2(newWidth, newHeight);

  int YD = (oldHeight / newHeight - 1) * oldWidth;
  int YR = oldHeight % newHeight;
  int XD = oldWidth / newWidth;
  int XR = oldWidth % newWidth;
  int outOffset = 0;
  int inOffset = 0;

  for (int y = newHeight, YE = 0; y > 0; y--) {
   for (int x = newWidth, XE = 0; x > 0; x--) {
    outputData.pixels[outOffset++] = inputData.pixels[inOffset];
    inOffset += XD;
    XE += XR;
    if (XE >= newWidth) {
     XE -= newWidth;
     inOffset++;
    }
   }
```

```
    inOffset += YD;
    YE += YR;
    if (YE >= newHeight) {
      YE -= newHeight;
      inOffset += oldWidth;
    }
  }

  image(outputData, xPos, yPos);
}
```

Animation Tab

```
////////////////////////////////////////////////////////////////////////////////////////
/////

// Animation Functions, Provide next value given the current value, target value, and speed

////////////////////////////////////////////////////////////////////////////////////////
/////


// linear animation function (relative)
// e.g. easeIn(x, 100, 2);
int animateTo(int current, int target, int speed)
{
  int next = current;          // initially our next step is where we currently are
  if(target < next){           // if we need to make our value smaller...
    next = next - speed;       // make it smaller
    next = max(next, target);  // ensure that we didn't make it smaller than our target value
  }
  else if(next < target){      // if we need to make our value bigger...
    next = next + speed;       // make it bigger
    next = min(next, target);  // ensure that we didn't make it bigger than our target value
  }
```

```
  return next;                // return our next value
}

// ease out animation function (relative)
// e.g. easeIn(x, 100, 10);
int easeOut(int currentOriginal, int targetOriginal, int speed)
{
  // make values large for math
  int current = currentOriginal * 1000; // make current value large for math (e.g. 2 becomes 2000)
  int target = targetOriginal * 1000;   // make target value large for math  (e.g. 33 becomes 33000)

  // do math to calculate our next value
  int change = target - current;     // find out how much change there is (e.g. 33000 - 2000 = 31000)
  int changeLittle = change / speed; // make the change a little change   (e.g. 31000 / 4 = 7750)
  int next = current + changeLittle; // change the current value a little (e.g. 2000 + 7750 = 9750)

  // make next value small for screen
  next = next / 1000;

  // if our little change was so little that we didn't move...
  if(next == currentOriginal){
    next = targetOriginal; // our next step is our target
  }

  return next; // return our next value (e.g. 9750 / 1000 = 9, remember that we started with 2)
}

// eased in animation function (relative), somewhat hack but ok for simple animation
// e.g. easeIn(x, 100, 900);
int easeIn(int current, int target, int speed)
```

```
{
  int change = target - current; // record how far we are away
  int next = current;           // initially our next step is where we currently are
  if(target < next){           // if we need to make our value smaller...
    change = target - change;    // flip how far we are away
    speed += change;            // divide by zeroes are bad
    speed /= change;            // divide
    next = next - speed;        // make it smaller
    next = max(next, target);    // ensure that we didn't make it smaller than our target value
  }
  else if(next < target){      // if we need to make our value bigger...
    speed += change;           // divide by zeroes are bad
    speed /= change;           // divide
    next = next + speed;       // make it bigger
    next = min(next, target);  // ensure that we didn't make it bigger than our target value
  }
  return next;                // return our next value
}
```