

TYPE+CODE IV WORKSHOP

WHAT IS PROCESSING?

PROCESSING IS A DIGITAL TOOL, AN ON-LINE COMMUNITY, A TEACHING ENVIRONMENT, AND A POWERFUL MEDIUM OF VISUAL EXPRESSION. CREATED BY BEN FRY AND CASEY REAS, GRADUATES OF MIT'S MEDIA LAB, PROCESSING IS THE LANGUAGE BEHIND A BODY OF VISUAL PROJECTS PRODUCED BY ARTISTS AND DESIGNERS AROUND THE WORLD. THIS OPEN-SOURCE APPLICATION CAN BE DOWNLOADED FOR FREE AT [HTTP://WWW.PROCESSING.ORG](http://www.processing.org), A WEB SITE WHERE A GLOBAL NETWORK OF USERS MEETS TO SHAPE THE LIFE OF THE PROGRAM AND SHARE THE RESULTS OF THEIR WORK.

WITH ITS DIRECT SYNTAX AND ELEGANT INTERFACE, PROCESSING ENABLES USERS WHO HAVE MINIMAL PROGRAMMING EXPERIENCE TO CREATE SIMPLE, RULE-BASED ANIMATIONS AND INTERACTIVE OR SELF-EVOLVING WORKS. WHEREAS MOST COMMERCIAL ANIMATION PROGRAMS (SUCH AS FLASH) EMPLOY A FAMILIAR PALETTE OF DRAWING TOOLS (BRUSH, PEN, PAINT BUCKET) AND A GRAPHIC TIMELINE FOR ORGANIZING ELEMENTS SEQUENTIALLY, DRAWING IN PROCESSING IS ENTIRELY CODE-BASED—REQUIRING THE DESIGNER TO WRITE INSTRUCTIONS FOR GENERATING A MARK ON THE SCREEN AND DEFINING ITS BEHAVIOR. THE APPARENT SIMPLICITY OF PROCESSING IS NO STOPPING POINT, HOWEVER, AS THE APPLICATION SUPPORTS AMBITIOUS INTERACTIVE PIECES AND INTRICATE SELF-GENERATING COMPOSITIONS. PROFESSIONAL DESIGNERS ARE USING THE PROGRAM AS PART OF THEIR RESEARCH AND PRODUCTION PROCESS, FROM THE ARCHITECTS MORPHOSIS IN SANTA MONICA, CALIFORNIA, TO THE INTERACTIVE MEDIA DESIGNERS ART+COM IN BERLIN, GERMANY. WORKING WITH SCIENTISTS, FRY IS USING PROCESSING TO VISUALIZE THE HUMAN GENOME AT THE BROAD INSTITUTE OF MIT AND HARVARD. AS ARTISTS, BOTH FRY AND REAS USE PROCESSING TO CREATE COMPLEX AUTOMATED DRAWINGS. REAS'S PROJECT ARTICULATE GENERATES A FIELD OF ELEMENTS THAT SPONTANEOUSLY INTERACT, RESEMBLING A CLOUD OF SPORES OR A BED OF LICHEN—SHRINKING, GROWING, MERGING, CLUSTERING, DISINTEGRATING. FRY'S DISARTICULATE TAKES WRITTEN CODE FROM REAS'S PIECE AND TRACES ITS REPETITIVE ROUTINES WITH DENSE FURLS OF MACHINE-DRAWN LINES. THESE TWO WORKS REFLECT A NEW ROLE FOR THE DESIGNER: TO DEFINE A POINT OF ORIGIN AND A SET OF CONDITIONS, AND THEN STEP BACK TO WATCH LIFE TAKE OVER.

WRITTEN BY ELLEN LUPTON, DESIGN LIFE NOW: NATIONAL DESIGN TRIENNIAL, SMITHSONIAN COOPER-HEWITT, NATIONAL DESIGN MUSEUM, P.166, 2006.

TYPE+CODE

STUDENTS HAVE BECOME ACCUSTOMED TO SOLVING DESIGN PROBLEMS THROUGH COMPLEX COMMERCIAL SOFTWARE PACKAGES THAT WILL EVOLVE RAPIDLY AND POSSIBLY DISAPPEAR IN THE NEAR FUTURE. HOW CAN WE PROVIDE STUDENTS WITH THE CONFIDENCE AND BROAD STRUCTURAL UNDERSTANDING THEY WILL NEED TO EDUCATE THEMSELVES AS THEIR FIELD CHANGES? FORMER MIT MEDIA LAB COLLABORATORS CASEY REAS AND BENJAMIN FRY PIONEERED THE OPEN-SOURCE PROJECT PROCESSING IN 2001. DESIGNED TO ENCOURAGE LEARNING CODE THROUGH EASY AND FREQUENT VISUAL FEEDBACK, PROCESSING IS A SIMPLE BUT DECEPTIVELY POWERFUL PROGRAMMING LANGUAGE THAT CAN GENERATE STARTLING VISUAL EFFECTS. THROUGH THE APPLICATION OF BASIC MATHEMATICAL CONCEPTS (INCLUDING RANDOM PROCESSES AND RULE-BASED SYSTEMS), UNEXPECTED EXPRESSIONS THAT MIGHT TAKE DAYS TO CREATE BY HAND CAN BE GENERATED IN SECONDS. VIRTUALLY ANY TYPE OF DATA SET -- FROM SOUND AND OTHER "CAPTURED" ACTIVITY TO RFID TAGS AND BLOGS -- CAN BE USED TO GENERATE WORK THAT IS NOT BOUND TO THE COMPUTER SCREEN OR TO PRINT. PROCESSING USERS ARE FINDING NEW WAYS TO USE THIS FLEXIBILITY EVERY DAY, SENDING THEIR INTERPRETED DATA TO OBJECTS AS VARIED AS DRAWING MACHINES, ARCHITECTURAL FACADES, AND CELL PHONES. LEARNING TO WORK WITH CODE CAN BE AS FUNDAMENTAL TO THE DESIGNER'S EDUCATION AS LEARNING TO BIND A BOOK OR PRINT WITH LETTERPRESS, PARTICULARLY FOR THOSE WHO WISH TO WORK WITH NON-TRADITIONAL MEDIA. BY LEARNING TO PERFORM BASIC OPERATIONS DIRECTLY IN A PROGRAMMING LANGUAGE, STUDENTS ARE EXPOSED TO THE CORE STRUCTURES THAT UNDERLIE THE HIGH-LEVEL TOOLS USED IN THE PROFESSION, WHILE ALSO EXPANDING THEIR ABILITIES AND EXPERIENCE IN NEW MEDIA. YEOHYUN AHN IS DEVELOPING A SET OF ON-LINE RESOURCES AND TEACHING TOOLS CREATED ESPECIALLY FOR DESIGNERS AND DESIGN STUDENTS WITH LIMITED PRIOR KNOWLEDGE OF COMPUTER LANGUAGES. THEY ARE BUILDING TUTORIALS AROUND BASIC DESIGN OPERATIONS SUCH AS REPEAT, ROTATE, MOVE, INVERT, CUT, AND RANDOM AS WELL AS GRAPHIC DESIGN FUNCTIONS SUCH AS TRANSPARENCY, LAYER, COLOR, HIERARCHY, FIGURE/GROUND.

PRESENTED BY YEOHYUN AHN AND GREGORY MAY AT SCHOOL OF THOUGHTIII IN 2007 AT ART CENTER COLLEGE OF DESIGN, PASADENA, CA.

RECOMMEND WEBSITES

[HTTP://WWW.PROCESSING.ORG](http://www.processing.org)
[HTTP://WWW.TYPEANDCODE.COM](http://www.typeandcode.com)
[HTTP://WWW.REAS.COM](http://www.reas.com)
[HTTP://WWW.BENFRY.COM/](http://www.benfry.com/)
[HTTP://WWW.EVOLUTIONZONE.COM/](http://www.evolutionzone.com/)
[HTTP://WWW.ABSTRACT-CODEX.NET/WORKS.HTML](http://www.abstract-codex.net/works.html)
[HTTP://INCUBATOR.QUASIMONDO.COM/](http://incubator.quasimondo.com/)
[HTTP://TYPOTOPO.COM/ABOUT.PHP?SHOW=LINKS](http://typotopo.com/about.php?show=links)
[HTTP://WWW.TYPEANDCODE.COM](http://www.typeandcode.com)
[HTTP://WWW.BENFRY.COM/](http://www.benfry.com/)

RECOMMENDED FOR TEACHERS

[HTTP://WWW.SHIFFMAN.NET/TEACHING/THE-NATURE-OF-CODE/](http://www.shiffman.net/teaching/the-nature-of-code/)

RECOMMENDED PAPERS

COMPUTATIONAL MODELS FOR EXPRESSIVE DIMENSIONAL TYPOGRAPHY, PETER CHO
COMPUTATIONAL INFORMATION DESIGN, BEN FRY

HOW TO START WITH PROCESSING?

DOWNLOAD: [HTTPS://WWW.PROCESSING.ORG/DOWNLOAD](https://www.processing.org/download)
REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/](http://www.processing.org/reference/)

PROCESSING LIBRARIES

EXTENDING PROCESSING BEYOND GRAPHICS AND IMAGES, LIBRARIES ENABLE TO COMMUNICATE WITH OTHER DEVICES SUCH AS AUDIO AND VIDEO. FOR INSTANCE, REAL TIME SOUND INPUT, MOTION TRACKING WITH WEB CAMERA, ETC.

[HTTP://WWW.PROCESSING.ORG/REFERENCE/LIBRARIES/](http://www.processing.org/reference/libraries/)

HOW TO DRAW LINE, CIRCLE, RECTANGLE AND CURVE?

BEFORE WE START, I WOULD LIKE TO TALK ABOUT A CODE STRUCTURE IN PROCESSING. THERE ARE MANY WAYS THAT YOU CAN ORGANIZE YOUR FUTURE CODES. FOR ME, I WILL USE THE FOLLOWINGS IN PROCESSING:

```
VOID SETUP()
{
}
```

```
VOID DRAW()
{
}
```

WHAT IS SETUP?

SETUP() IS USED TO DEFINE YOUR INITIAL ENVIRONMENT SUCH AS SCREEN SIZE AND BACKGROUND COLOR. YOU SHOULD HAVE ONLY ONE SETUP() IN YOUR CODE.
REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/SETUP_.HTML](http://www.processing.org/reference/setup_.html)

WHAT IS DRAW?

DRAW() IS USED TO AFTER SETUP(). YOU CAN INCLUDE ANY DRAWING FUNCTION SUCH AS LINE(), RECT(), ELLIPSE(), CURVE(), BEZIER(), ETC.
REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/DRAW_.HTML](http://www.processing.org/reference/draw_.html)

HOW TO DRAW LINE?

REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/LINE_.HTML](http://www.processing.org/reference/line_.html)

```
VOID SETUP()
{
  SIZE(500,500);
}
```

```
VOID DRAW()
{
  LINE(30, 20, 85, 75);
}
```

HOW TO DRAW RECTANGLE:

REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/RECT_.HTML](http://www.processing.org/reference/rect_.html)

```
VOID SETUP()
{
  SIZE(500,500);
}
```

```
VOID DRAW()
{
  RECT(30, 20, 55, 55);
}
```

HOW TO DRAW CIRCLE:

REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/ELLIPSE_.HTML](http://www.processing.org/reference/ellipse_.html) // EXERCISE 3

```
VOID SETUP()
{
  SIZE(500,500);
}
```

```
VOID DRAW()
{
  ELLIPSE(56, 46, 55, 55);
}
```

IF YOU WOULD WANT TO APPLY ANY SPECIFIC STROKE WEIGHT ON YOUR CIRCLE, YOU CAN USE STROKEWEIGHT();

REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/STROKEWEIGHT_.HTML](http://www.processing.org/reference/strokeweight_.html)

```
VOID SETUP()
{
  SIZE(500,500);
}
```

```
VOID DRAW()
{
  STROKEWEIGHT(5);
  ELLIPSE(56, 46, 55, 55);
}
```

IF YOU WOULD WANT TO APPLY ANY SPECIFIC STROKE WEIGHT ON YOUR CIRCLE, YOU CAN USE STROKEWEIGHT();
REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/STROKEWEIGHT_.HTML](http://www.processing.org/reference/strokeweight_.html)

```
VOID SETUP()
{
  SIZE(500,500);
}
```

```
VOID DRAW()
{
  STROKEWEIGHT(5);
  ELLIPSE(56, 46, 55, 55);
}
```

IF YOU WOULD WANT TO ADD A SPECIFIC STROKE COLOR IN YOUR STROKE, YOU CAN USE STROKED();
REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/STROKE_.HTML](http://www.processing.org/reference/stroke_.html)

```
VOID SETUP()
{
  SIZE(500,500);
}
VOID DRAW()
{
  STROKEWEIGHT(10);
  STROKE(255,0,0);
  ELLIPSE(56, 46, 55, 55);
}
```

IF YOU WOULD WANT TO CREATE A CIRCLE WITH THE FOLLOWING OPTIONS,

- WHITE BACKGROUND
- BLACK COLOR
- STROKE COLOR(RED)
- STROKE WEIGHT(5 PIXEL)

```
VOID SETUP()
{
  SIZE(500,500);
  BACKGROUND(255,255,255);
}
```

```
VOID DRAW()
{
  STROKEWEIGHT(5);
  STROKE(255,0,0);
  FILL(0,0,0);
  ELLIPSE(56, 46, 55, 55);
}
```

IF YOU WOULD WANT TO CREATE NUMEROUS CIRCLES BY USING YOUR MOUSE INTERACTIVITY, YOU CAN USE MOUSEX AND MOUSEY.
MOUSEX: [HTTP://WWW.PROCESSING.ORG/REFERENCE/MOUSEX.HTML](http://www.processing.org/reference/mousex.html)
MOUSEY: [HTTP://WWW.PROCESSING.ORG/REFERENCE/MOUSEY.HTML](http://www.processing.org/reference/mousey.html)

```

VOID SETUP()
{
  SIZE(500,500);
  BACKGROUND(255,255,255);}

```

```

VOID DRAW()
{
  STROKEWEIGHT(1);
  STROKE(255,0,0);
  NOFILL();
  ELLIPSE(MOUSEX, MOUSEY, 100, 100);
}

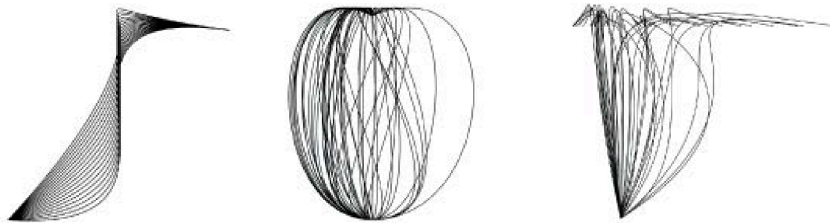
```

WHAT IS BEZIERO?

IT ENABLES TO DRAW A BEZIER CURVE ON YOUR SCREEN. IT IS DEFINED BY A SERIES OF ANCHOR AND CONTROL POINT. IF YOU KNOW ADOBE ILLUSTRATOR'S PEN TOOL, YOU COULD DEFINITELY UNDERSTAND IT.

REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/BEZIER_.HTML](http://www.processing.org/reference/bezier_.html)

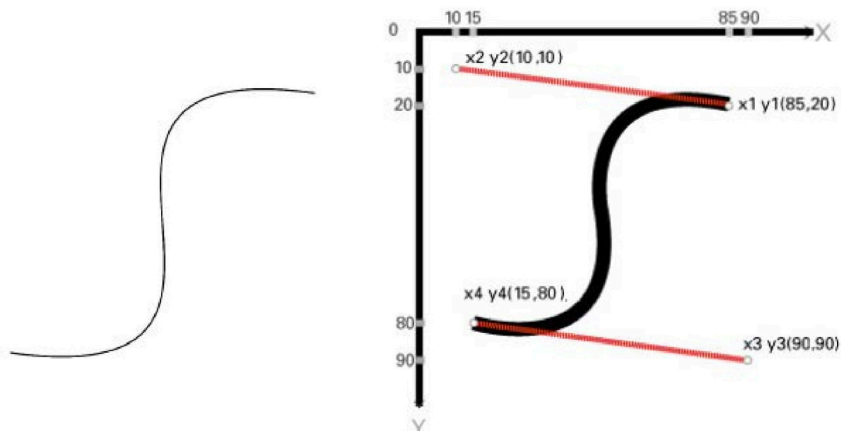
HERE IS SOME SAMPLES BY USING BEIZERO AND FORO STATEMENT.



```

VOID SETUP()
{
  SIZE(100,100);
  BACKGROUND(255,255,255);
}
VOID DRAW()
{
  BEZIER(85, 20, 10, 10, 90, 90, 15, 80);
}

```



```

VOID SETUP()
{
  SIZE(100,100);
  BACKGROUND(255);
  NOLOOP();
}

VOID DRAW()
{
  STROKE(0,0,0);
  STROKEWEIGHT(1);
  BEZIER(85,20,10,10,90,90,15,80);
}

```

NEXT IS WE WILL INCLUDE FOR() STATEMENT IN THIS CURRENT CODE:

WHAT IS FOR() STATEMENT?

[HTTP://WWW.PROCESSING.ORG/REFERENCE/FOR.HTML](http://www.processing.org/reference/for.html)

I ADDED A FOR() STATEMENT WITH THE CURRENT CODE LIKE THE FOLLOWING:

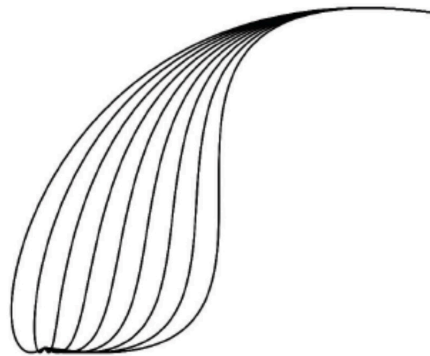
```

VOID SETUP()
{
  SIZE(100,100);
  BACKGROUND(255,255,255);
  NOLOOP();
}

VOID DRAW()
{
  NOFILL();
  STROKEWEIGHT(1);
  STROKE(0,0,0);
  SMOOTH();
  FOR(INT I=0; I<90; I+=10)
  {
    BEZIER(85, 20, 10, 10, I, 90, 15, 80);
  }
}

```

Output



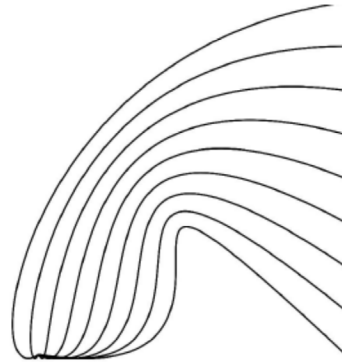
```

VOID SETUP()
{
  SIZE(100,100);
  BACKGROUND(255,255,255);
  NOLOOP();
}

```

Output

```
}  
VOID DRAW()  
{  
  NOFILL();  
  STROKEWEIGHT(1);  
  STROKE(0,0,0);  
  SMOOTH();  
  FOR(INT I=0; I<90; I+=10){  
    BEZIER(85, I, 10, 10, I, 90, 15, 80);  
  }  
}
```



```
VOID SETUP()  
{  
  SIZE(100,100);  
  BACKGROUND(255,255,255);  
  NOLOOP();  
}  
VOID DRAW()  
{  
  NOFILL();  
  STROKEWEIGHT(1);  
  STROKE(0,0,0);  
  SMOOTH();  
  FOR(INT I=0; I<90; I+=4){  
    BEZIER(I, 20, I, 10, 90, 90, 15, 80);  
  }  
}
```

```
VOID SETUP()  
{  
  SIZE(100,100);  
  BACKGROUND(255,255,255);  
  NOLOOP();  
}  
VOID DRAW()  
{  
  NOFILL();  
  STROKEWEIGHT(0.3);  
  STROKE(0,0,0);  
  SMOOTH();  
  FOR(INT I=0; I<90; I+=5)  
  {
```

```
BEZIER(90, 20, 10, 10, 90, 90, 1, 80);  
}  
}
```

```
VOID SETUP()  
{  
  SIZE(100,100);  
  BACKGROUND(255,255,255);  
  NOLOOP();  
}  
VOID DRAW()  
{  
  NOFILL();  
  STROKEWEIGHT(0.3);  
  STROKE(0,0,0);  
  SMOOTH();  
  FOR(INT I=0; I<90; I+=5)  
  {  
    BEZIER(90, 20, 10, 10, 90, I, 5, 80);  
  }  
}
```

WE AS DESIGNERS OR DESIGN AND ART MAJORS, WE WOULD NEED TO CONVERT IT TO A PDF DOCUMENT. HOW TO SAVE AS PDF?

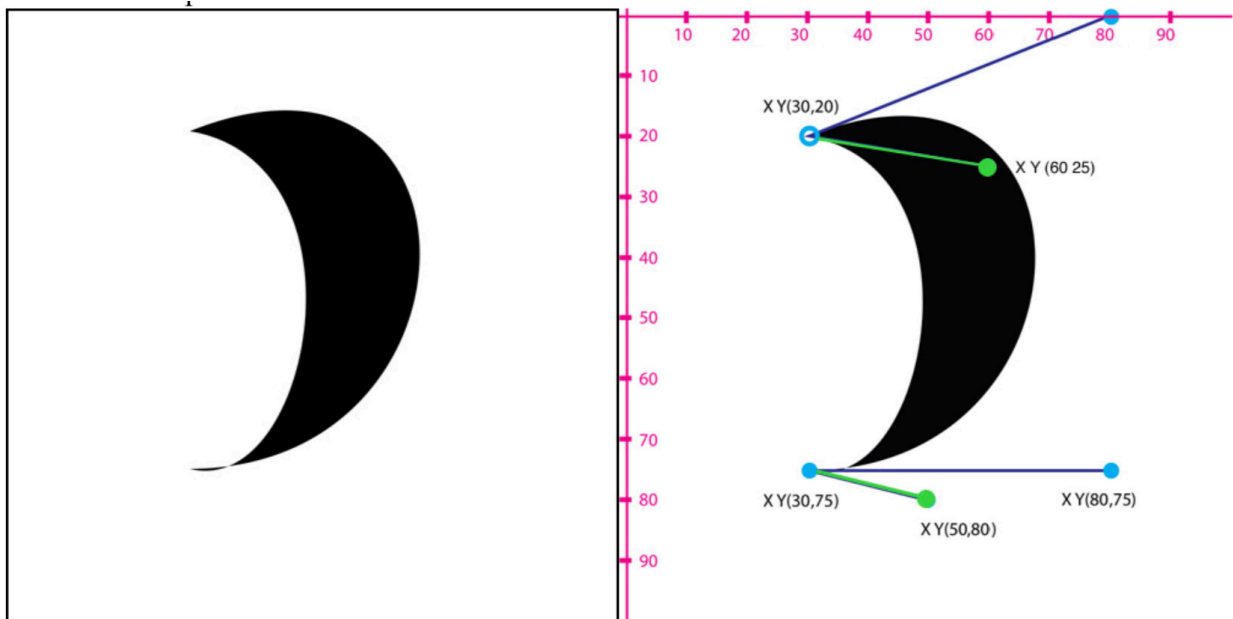
```
IMPORT PROCESSING.PDF.*;  
VOID SETUP()  
{  
  SIZE(100,100);  
  BEGINRECORD(PDF, "BEIZER.PDF");  
  BACKGROUND(255,255,255);  
  NOLOOP();  
}  
VOID DRAW()  
{  
  NOFILL();  
  STROKEWEIGHT(0.3);  
  STROKE(0,0,0);  
  SMOOTH();  
  FOR(INT I=0; I<90; I+=5)  
  {  
    BEZIER(90, 20, 10, 10, 90, I, 5, 80);  
  }  
  ENDRECORD;  
}
```


WHAT IS BEIZERVERTEXO?



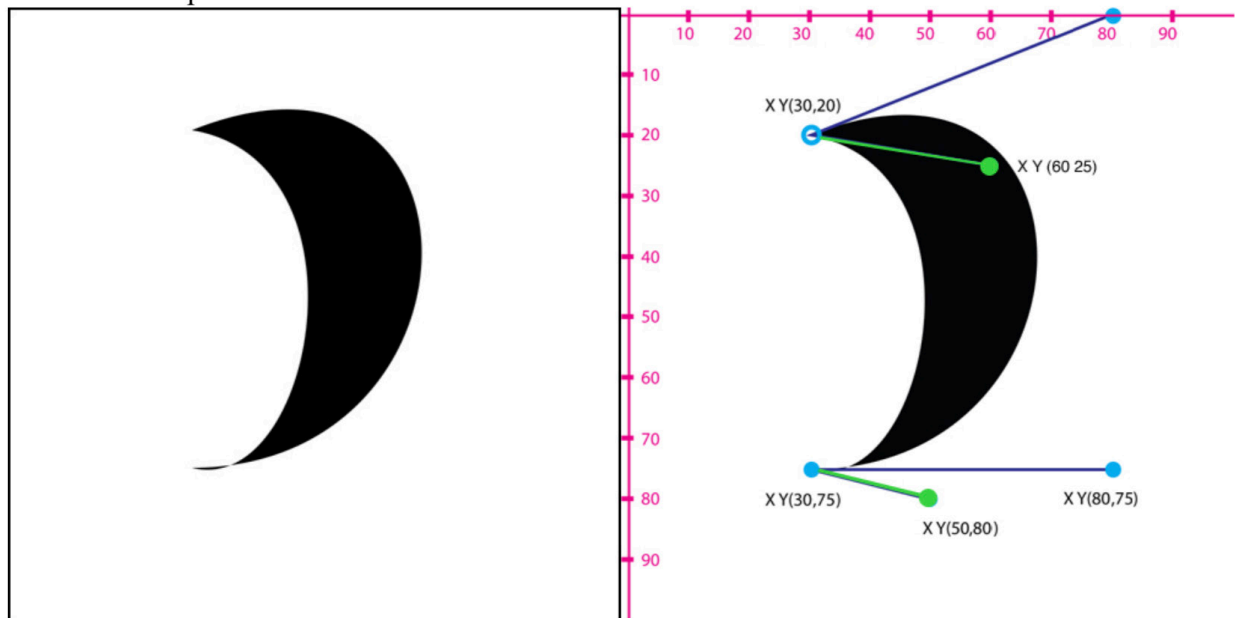
```
VOID SETUP(){  
  SIZE(100,100);  
  BACKGROUND(255,255,255);  
  NOLOOP();  
  
}  
VOID DRAW(){  
  FILL(0,0,0);  
  NOSTROKE();  
  BEGINSHAPE();  
  VERTEX(30, 20);  
  BEZIERVERTEX(80, 0, 80, 75, 30, 75);  
  BEZIERVERTEX(50, 80, 60, 25, 30, 20);  
  ENDSHAPE();  
}
```

HERE IS THE OUTPUT:



EXAMPLE:

```
BEGINSHAPE();  
VERTEX(30, 20);  
BEZIERVERTEX(80, 0, 80, 75, 30, 75);  
BEZIERVERTEX(50, 80, 60, 25, 30, 20);  
ENDSHAPE();
```



WHAT IS BEZIERVERTEX()?

BEZIERVERTEX(); DEFINES THE POSITION OF TWO CONTROL POINTS AND ONE ANCHOR POINT OF A BEZIER CURVE, ADDING A NEW SEGMENT TO A LINE OR SHAPE. THE FIRST TIME BEZIERVERTEX() IS USED WITHIN A BEGINSHAPE() CALL, IT MUST BE PREFACED WITH A CALL TO VERTEX() TO SET THE FIRST ANCHOR POINT. THIS FUNCTION MUST BE USED BETWEEN BEGINSHAPE() AND ENDSHAPE() AND ONLY WHEN THERE IS NO MODE PARAMETER SPECIFIED TO BEGINSHAPE().

REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/BEZIERVERTEX_.HTML](http://www.processing.org/reference/beziervertex_.html)

SYNTAX: BEZIERVERTEX(CX1, CY1, CX2, CY2, X, Y)

PARAMETERS:

CX1: X-COORDINATE OF 1ST CONTROL POINT

CY1: Y-COORDINATE OF 1ST CONTROL POINT

CX2: X-COORDINATE OF 2ND CONTROL POINT

CY2: Y-COORDINATE OF 2ND CONTROL POINT

X: X-COORDINATE OF ANCHOR POINT

Y: Y-COORDINATE OF ANCHOR POINT

WHAT IS VERTEXO?

VERTEXO IS USED TO SPECIFY THE VERTEX COORDINATES FOR POINTS, LINES, TRIANGLES, QUADS, AND POLYGONS AND IS USED EXCLUSIVELY WITHIN THE BEGIN-SHAPEO AND ENDSHAPEO FUNCTION.

REFERENCE: [HTTP://WWW.PROCESSING.ORG/REFERENCE/VERTEX_.HTML](http://www.processing.org/reference/vertex_.html)

SYNTAX: VERTEX(X,Y);

X X-COORDINATE OF THE VERTEX

Y Y-COORDINATE OF THE VERTEX

TRANSLATEO;

[HTTP://WWW.PROCESSING.ORG/REFERENCE/TRANSLATE_.HTML](http://www.processing.org/reference/translate_.html)

```
//EXAMPLE 1 OF TRANSLATEO;
```

```
VOID SETUP()
{
  SIZE(600,600);
  NOLOOPO;
}
```

```
VOID DRAWO
```

```
{
  STROKEWEIGHT(3);
  ELLIPSE(300,300, 200, 200);
}
```

```
//EXAMPLE 2 OF TRANSLATEO;
```

```
VOID SETUP()
{
  SIZE(600,600);
  NOLOOPO;
}
```

```
VOID DRAWO
```

```
{
  STROKEWEIGHT(3);
  SMOOTHO;
  TRANSLATE(100,100);
  ELLIPSE(300,300, 200, 200);
}
```

ROTATEO;

[HTTP://WWW.PROCESSING.ORG/REFERENCE/ROTATE_.HTML](http://www.processing.org/reference/rotate_.html)





```
//EXAMPLE  
VOID SETUP
```

```
//EXAMPLE 1 OF ROTATED;
VOID SETUP()
{
SIZE(600,600);
NOLOOP();
}
VOID DRAW()
{
STROKEWEIGHT(5);
SMOOTH();
TRANSLATE(300, 300);
LINE(0,0, 300, 300);
}
```

```
//EXAMPLE 2 OF ROTATED;
VOID SETUP()
{
SIZE(600,600);
NOLOOP();
}
```

```
VOID DRAW()
{
STROKEWEIGHT(5);
SMOOTH();
TRANSLATE(300, 300);
ROTATE(PI/2);
LINE(0,0, 300, 300);
}
```

REFERENCE

Angle	Degrees	Radians
	90°	$\pi/2$
	60°	$\pi/3$
	45°	$\pi/4$
	30°	$\pi/6$

IN-CLASS EXERCISE: EXPERIMENTAL DRAWINGS BY USING BEIZERVERTEXO.

1. YOUR SCREEN SIZE SHOULD BE 600 PIXEL BY 600 PIXEL.
2. CREATE EXPERIMENTAL DRAWINGS BY USING VERTEXO;, AND BEIZERVERTEXO;
3. USE ANY FUNCTION SUCH AS FOR() STATEMENT, RANDOM(), STROKE(), STROKEWEIGHT(), FILL(), ETC.
4. YOU CAN COLLABORATE WITH ANY GEOMETRIC SHAPE SUCH AS ELLIPSE();

// SAMPLE OUTPUT BY YEOHYUN AHN

```
VOID SETUP(){
SIZE(600,600);
BACKGROUND(35);
NOLOOP();
}
VOID DRAW(){
TRANSLATE(200,200);
NOSTROKE();
FOR(INT I=0;I<600;I+=10)
{
STROKE(RANDOM(255),RANDOM(200),100);
STROKEWEIGHT(RANDOM(10));
NOFILL();
BEGINSHAPE();
VERTEX(10,RANDOM(200));
BEZIERVERTEX(RANDOM(), 0, I, RANDOM(), 50, RANDOM());
BEZIERVERTEX(50, RANDOM(), I, 25, RANDOM(), 20);
ENDSHAPE();
ROTATE(360/PI*I);
}
ENDRECORD();
}
```

GEOMERATIVE LIBRARY

DOWNLOAD: [HTTP://WWW.RICARDMARXER.COM/GEOMERATIVE/](http://www.ricardmarxer.com/geomerative/)

HERE IS THE SAMPE CODE TO EMBED YOUR ILLUSTRATION INTO A TYPEFACE. THE CODE IS ORIGINALLY FROM TUTORIAL_07_HELLOWORLD_GETPOINTS.PDE.

```
IMPORT GEOMERATIVE.*;

// DECLARE THE OBJECTS WE ARE GOING TO USE, SO THAT THEY ARE ACESIBLE FROM SETUP() AND FROM DRAW()
RFont F;
RShape GRP;
RPoint[] POINTS;

VOID SETUP() {
  // INITILAIZE THE SKETCH
  SIZE(800, 800);
  FRAMERATE(24);

  // CHOICE OF COLORS
  BACKGROUND(255);
  FILL(255, 102, 0);
  STROKE(0);

  // VERY IMPORTANT: ALLWAYS INITIALIZE THE LIBRARY IN THE SETUP
  RG.INIT(THIS);

  // LOAD THE FONT FILE WE WANT TO USE (THE FILE MUST BE IN THE DATA FOLDER IN THE SKETCH FLODER), WITH THE SIZE 60 AND THE ALIGNMENT CENTER
  GRP = RG.GETTEXT("F", "TIMES.TTF", 700, CENTER);

  // ENABLE SMOOTHING
  SMOOTH();
}

VOID DRAW() {
  // CLEAN FRAME
  BACKGROUND(255);

  // SET THE ORIGIN TO DRAW IN THE MIDDLE OF THE SKETCH
  TRANSLATE(WIDTH/2, 3*HEIGHT/4+50);

  // DRAW THE GROUP OF SHAPES
  NOFILL();
  STROKE(0, 0, 200, 150);
  RG.SETPOLYGONIZER(RG.ADAPTATIVE);
  // GRP.DRAW();

  // GET THE POINTS ON THE CURVE'S SHAPE
  //RG.SETPOLYGONIZER(RG.UNIFORMSTEP);
  //RG.SETPOLYGONIZERSTEP(MAP(FLOAT(MOUSEY), 0.0, FLOAT(HEIGHT), 0.0, 1.0));

  RG.SETPOLYGONIZER(RG.UNIFORMLENGTH);
  //RG.SETPOLYGONIZERLENGTH(20);
```

```
RG.SETPOLYGONIZERLENGTH(MAP(MOUSEY, 0, HEIGHT, 3, 150));
POINTS = GRP.GETPOINTS();
STROKEWEIGHT(5);
// IF THERE ARE ANY POINTS
IF (POINTS != NULL) {
NOFILL();

STROKE(0, 200, 0);
BEGINSHAPE();
FOR (INT I=0; I<POINTS.LENGTH; I++) {
// VERTEX(POINTS[I].X, POINTS[I].Y);
}
ENDSHAPE();

FILL(0);
STROKE(0);
FOR (INT I=0; I<POINTS.LENGTH; I++) {
// ELLIPSE(POINTS[I].X, POINTS[I].Y, 5, 5);
PUSHMATRIX();
TRANSLATE(POINTS[I].X, POINTS[I].Y);
/* YOU WILL PUT YOUR OWN CODE HERE*/
POPMATRIX();
}
}
}
```

SAMPLE DOWNLOAD: [HTTP://WWW.YEOAHN.COM/DESIGNANDCHANGE/ALBUMS/TYPE-CODE-IV-1/](http://www.yeoahn.com/designandchange/albums/type-code-iv-1/)