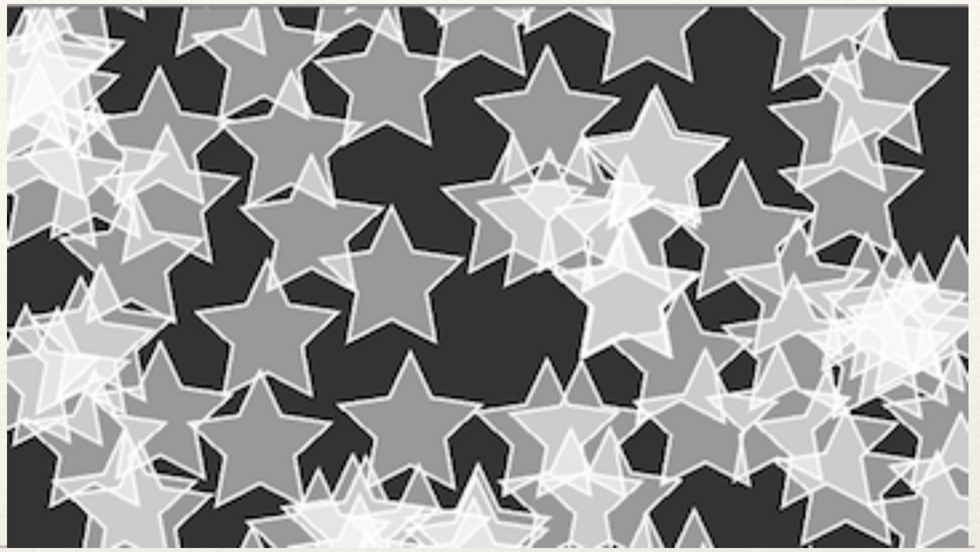
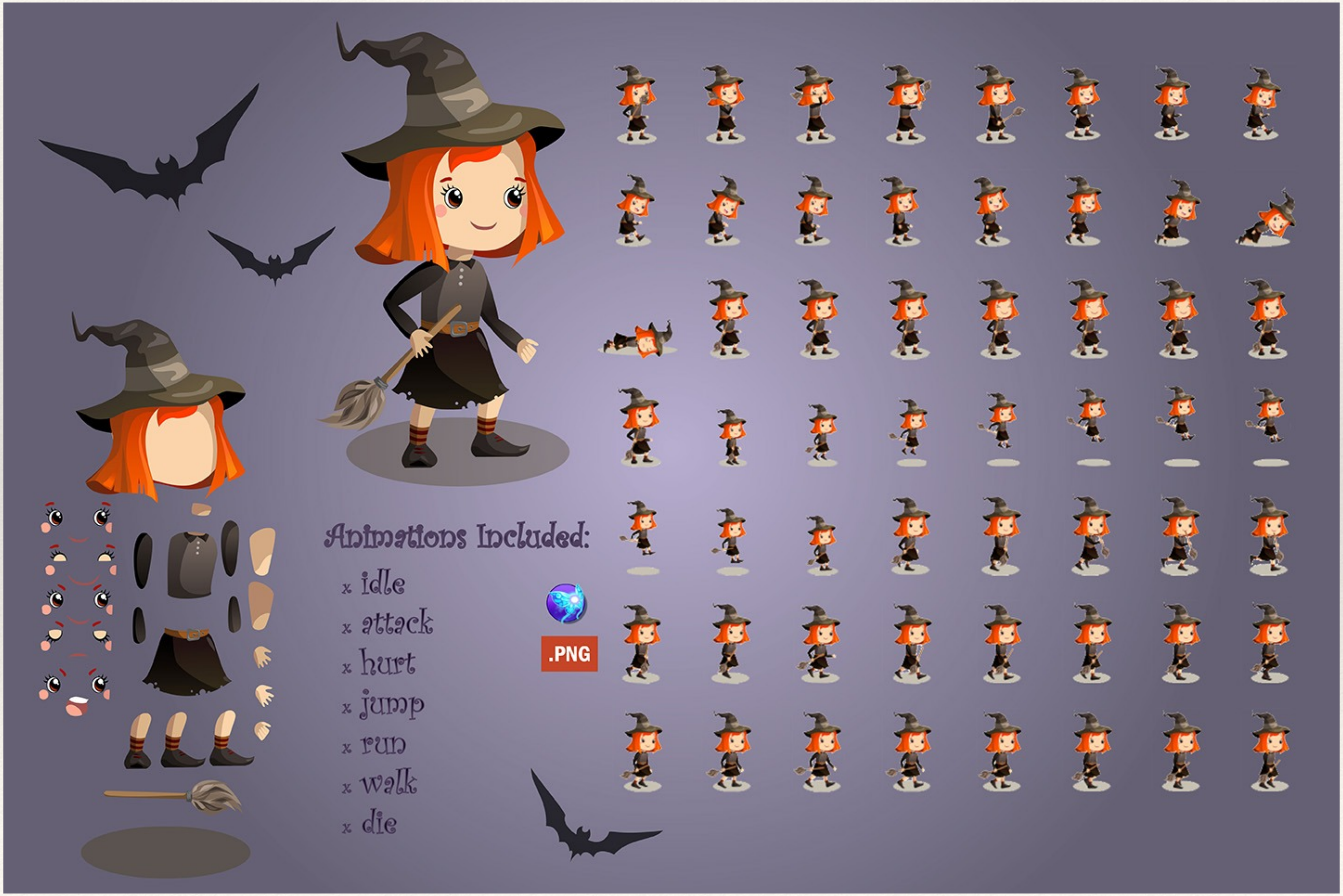
Dr. Sarah Abraham
University of Texas at Austin
Computer Science Department



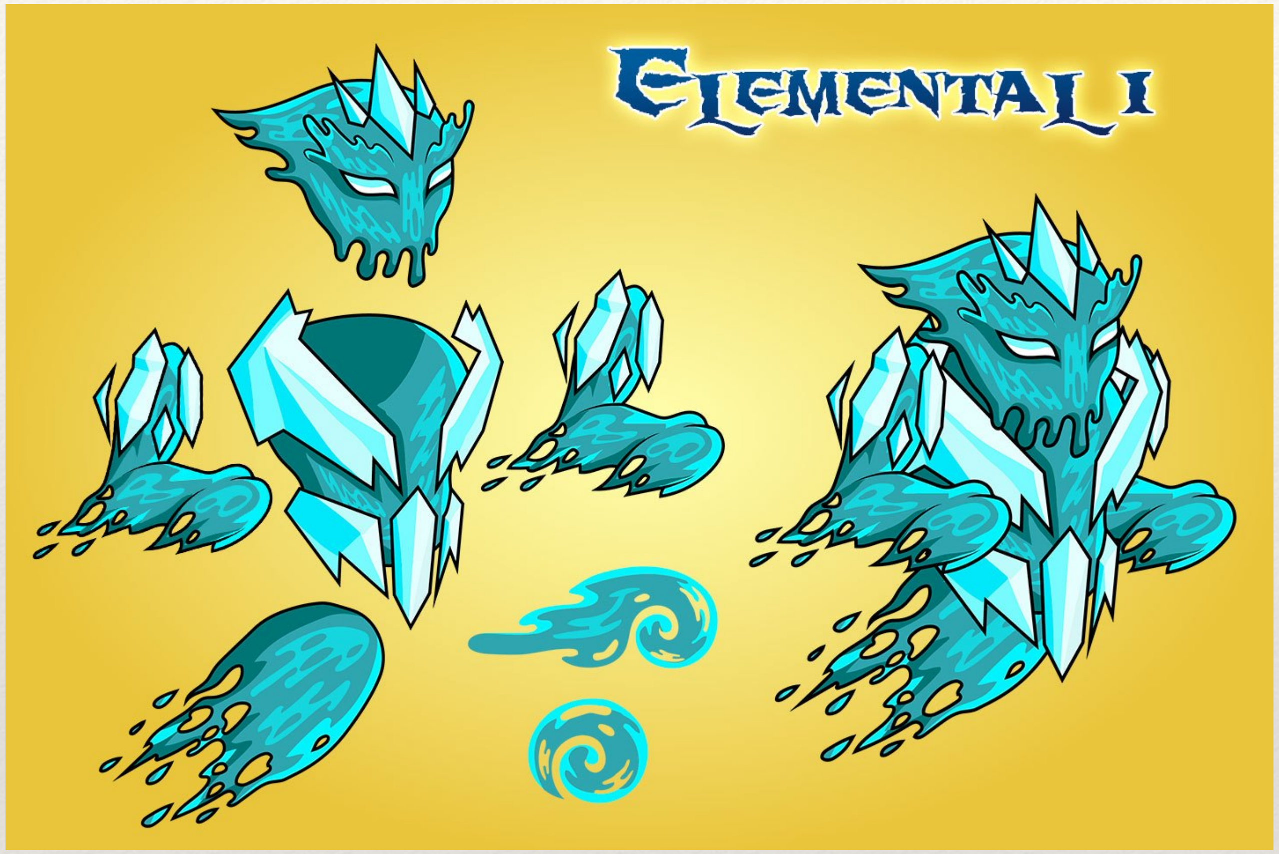
Shapes

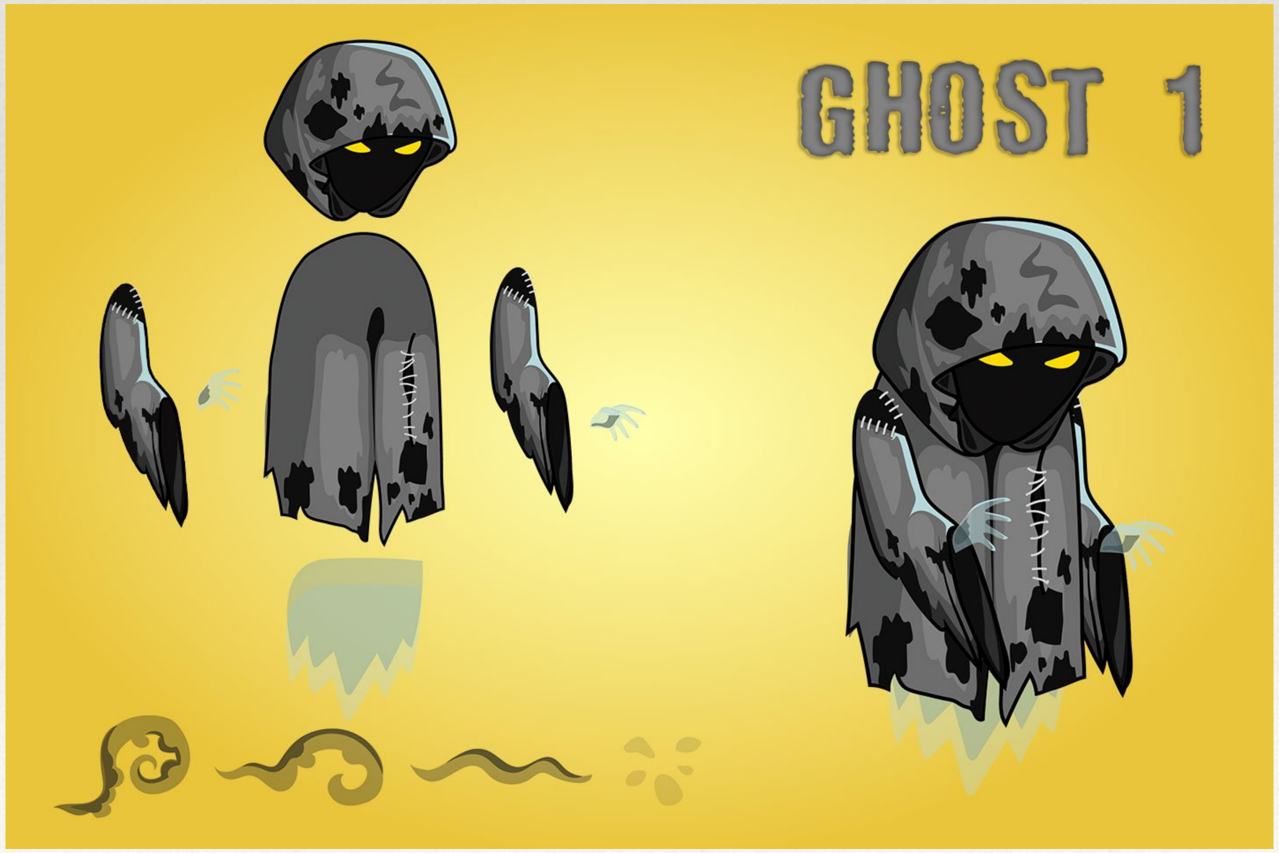
Elements of Graphics
CS324e

Animation Hierarchy Example



How to make the individual pieces?





Creating Shapes from Vertex Points

1. Use `beginShape()` function start shape
2. Specify points defining the shape with `vertex()`
3. Complete the shape with `endShape()`

- `fill()`, `stroke()`, `noFill()`, `noStroke()` and `strokeWeight()` control the shape attributes
- `endShape(CLOSE)` closes the shape

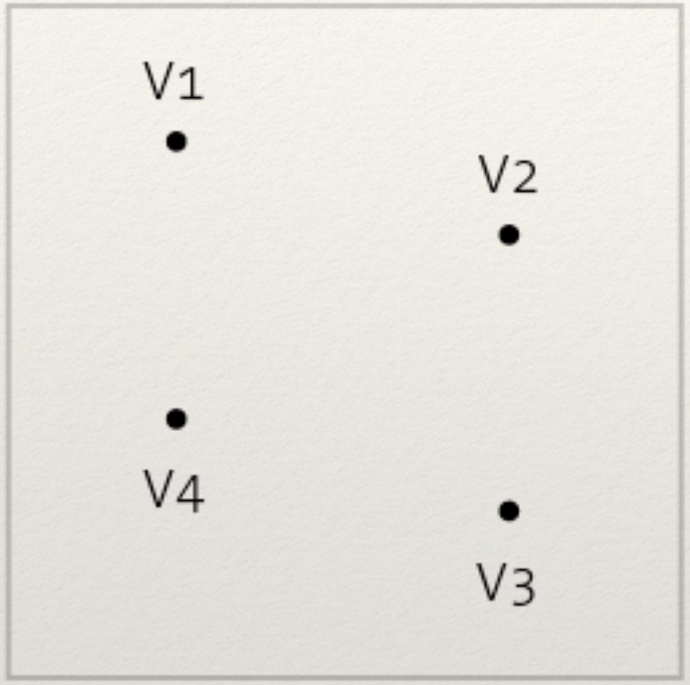
Vertex Points Example

```
beginShape();

vertex(30, 20);

vertex(85, 20);

vertex(85, 75);

vertex(30, 75);

endShape();

//endShape(CLOSE) would add line connecting back to the first vertex
```

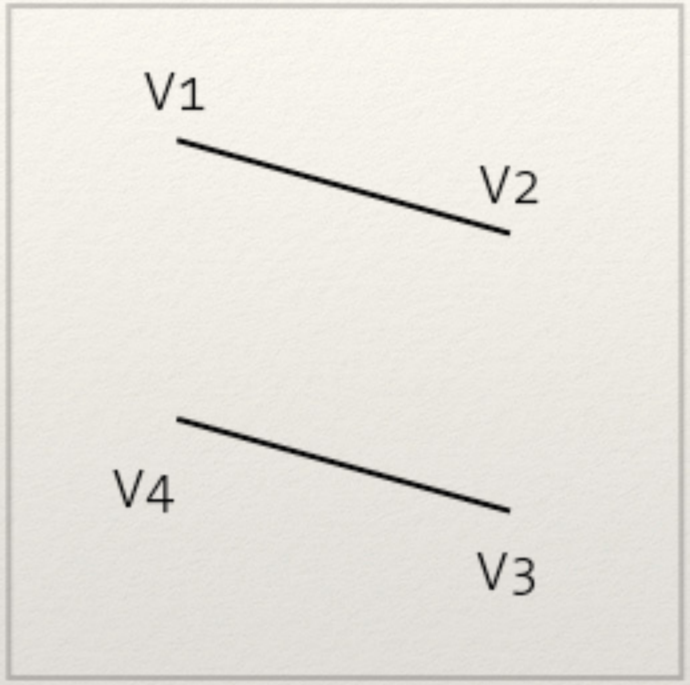
Geometry

- `beginShape()` accepts different parameters to define drawing of vertex data
    - POINTS
    - LINES
    - TRIANGLES
    - TRIANGLE_STRIP
    - TRIANGLE_FAN
    - QUADS
    - QUAD_STRIP

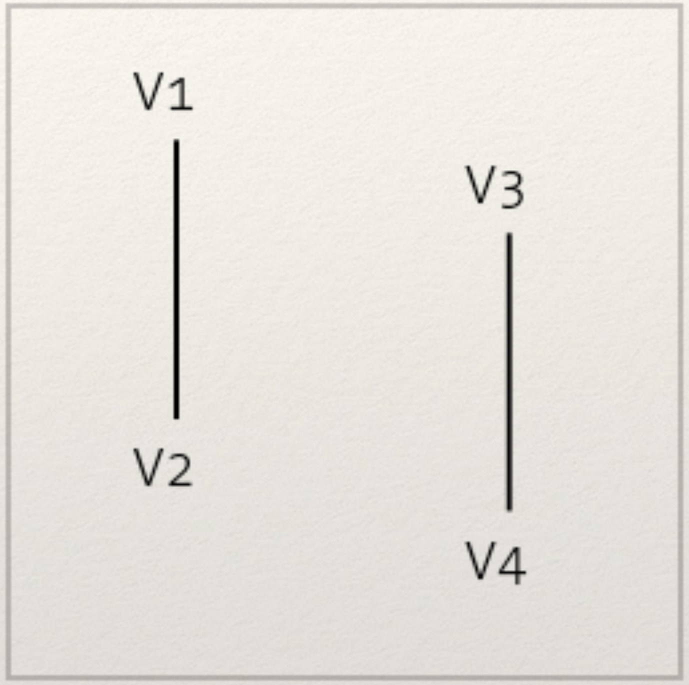
Geometry Examples



POINTS

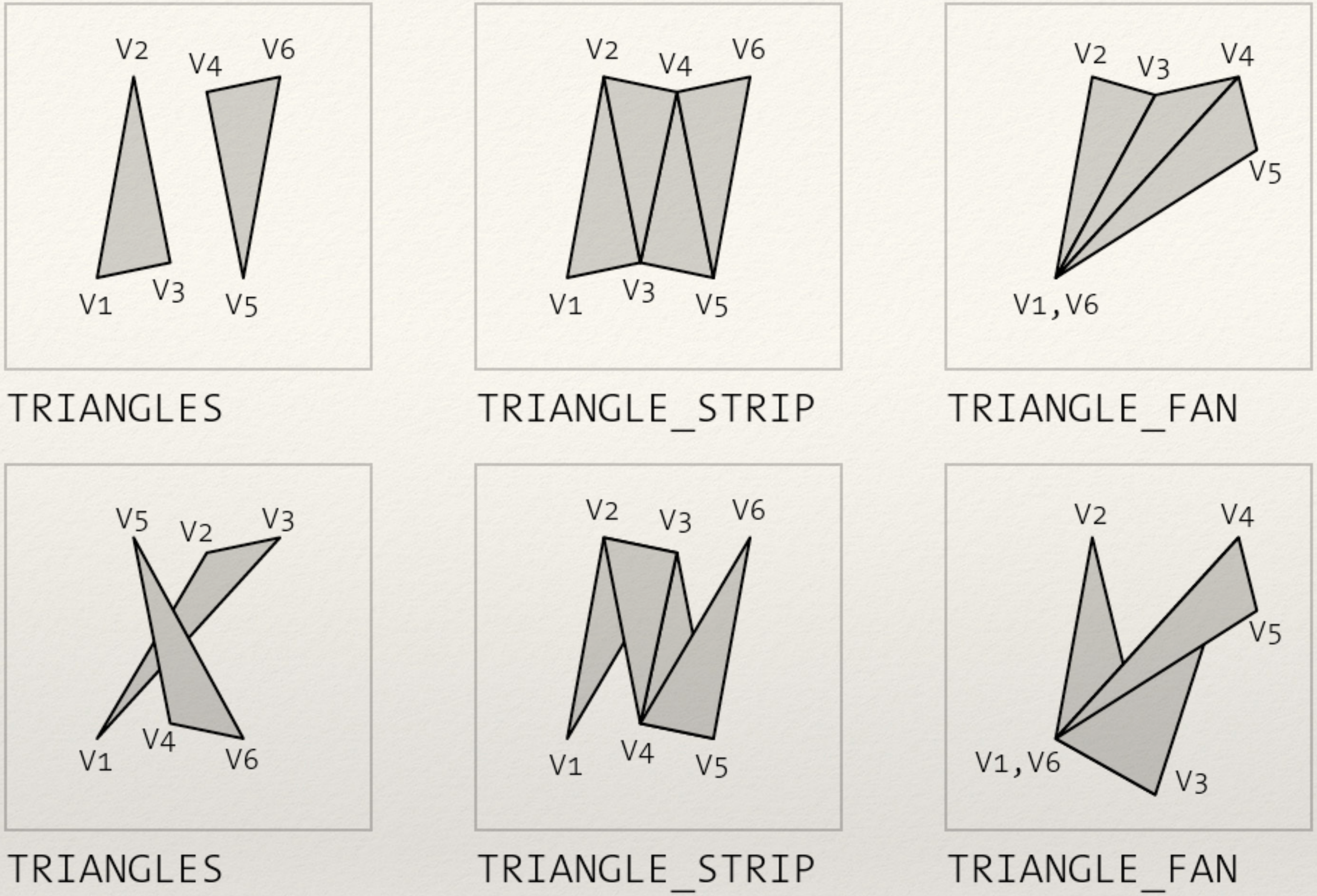


LINES



LINES

- `beginShape(POINTS);`
- `beginShape(LINES);`



TRIANGLES

TRIANGLE_STRIP

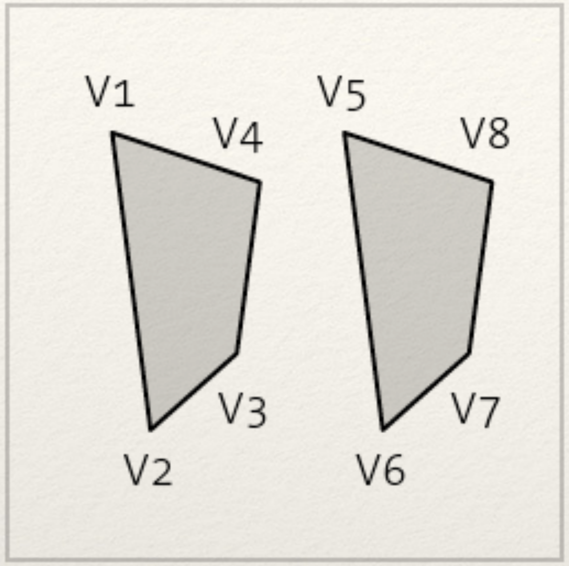
TRIANGLE_FAN

TRIANGLES

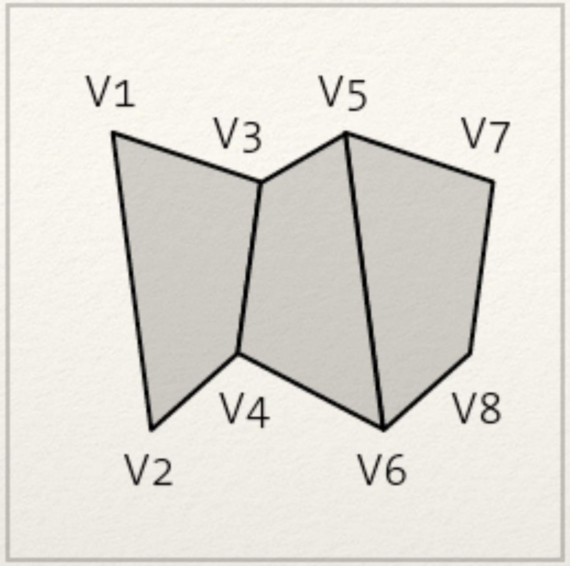
TRIANGLE_STRIP

TRIANGLE_FAN

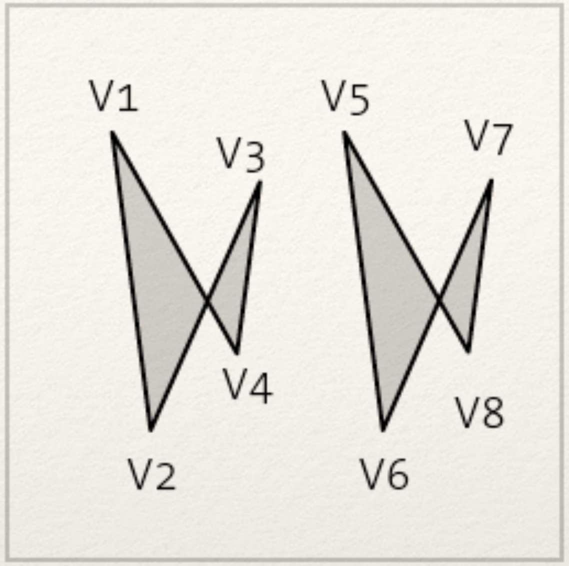
- beginShape(TRIANGLES);
- beginShape(TRIANGLE_STRIP);
- beginShape(TRIANGLE_FAN);



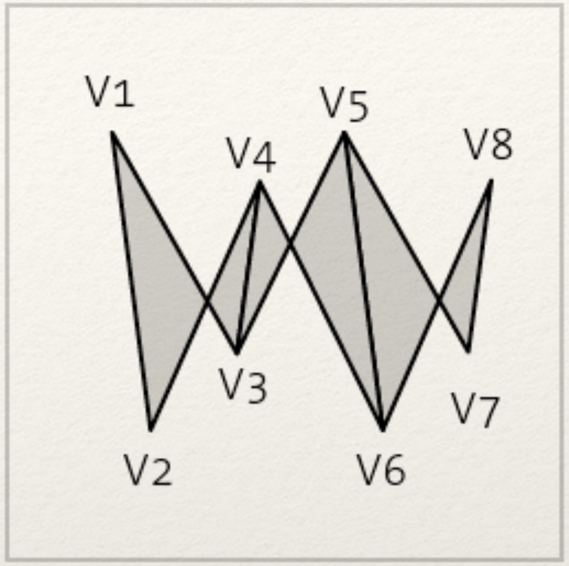
QUADS



QUAD_STRIP



QUADS



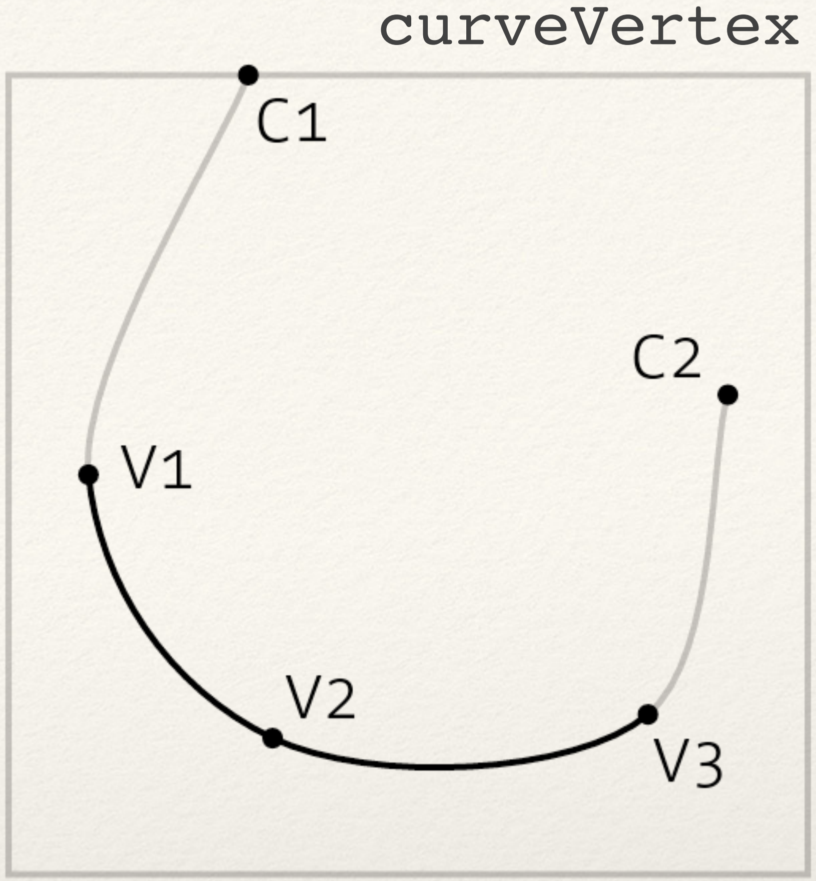
QUAD_STRIP

- beginShape(QUADS);
- beginShape(QUAD_STRIP);

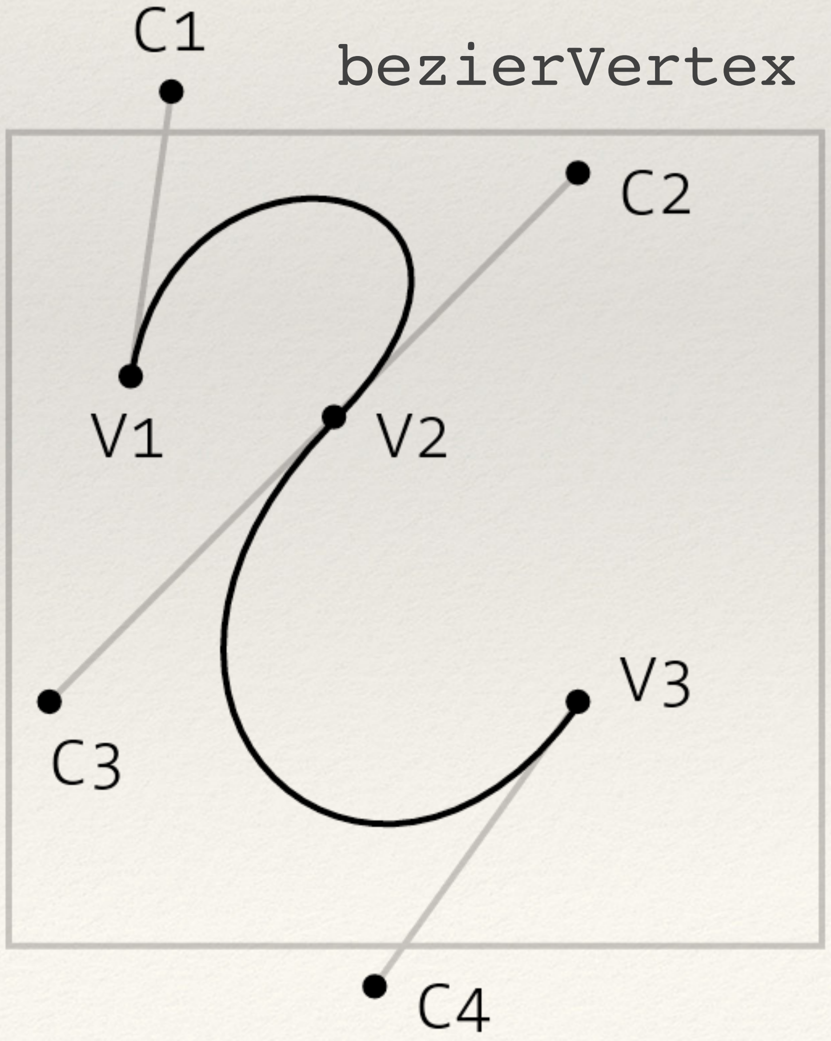
Curves

- `beginShape()` and `endShape()` can create curves if `beginShape()` has no parameter:
  - `curveVertex()`
  - `bezierVertex()`
- `curveVertex()` defined by first control point, intermediate vertices, and second control point
- `bezierVertex()` defined by initial anchor point then two control points with an anchor point

```
beginShape();
curveVertex(C1);
curveVertex(V1);
curveVertex(V2);
curveVertex(V3);
curveVertex(C2);
endShape();
```

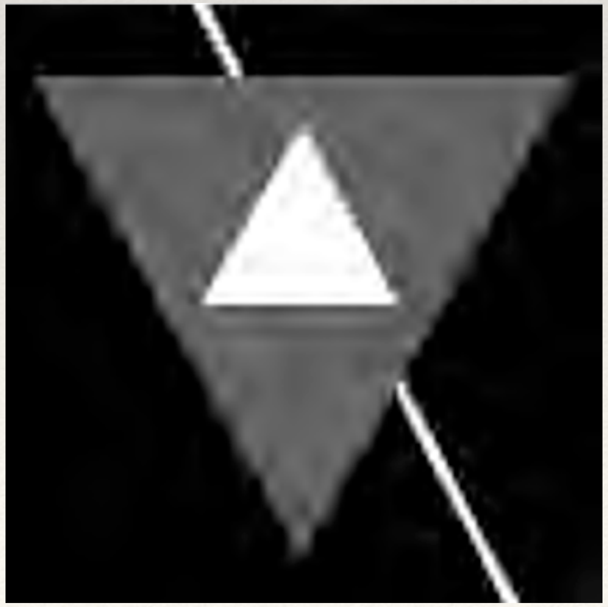


```
beginShape();
vertex(V1);
bezierVertex(C1, C2, V2);
bezierVertex(C3, C4, V3);
endShape();
```

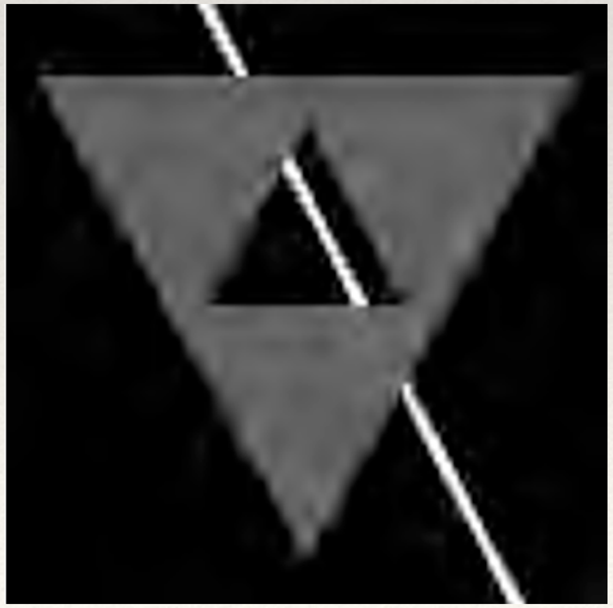


Contours

- Contours cut holes into Shapes
- `beginContour()`/`endContour()` called within `beginShape()`/`endShape()` code block
- `beginContour()` does not take parameters
- Must draw contour in **opposite winding direction** of shape!



Shape placed over Shape



Contour removing from Shape

Defining Shapes with Vertices

- A hard task
- Even harder defining 3D shapes
- What if we don't want to define curves by hand?
- What if we want to import shapes from vector programs like Illustrator or 3D modeling programs like Maya?

PShape

- Class for storing Shapes
- Loads and displays SVGs and OBJs
  - SVG is open standard for storing 2D vector graphics
  - OBJ is standard for storing 3D vector geometry
- `loadShape(“filename”)` to load into PShape
- `shape(PShape, x, y);` or `shape(PShape, x, y, width, height);` to display it

Load and Display SVGs

```
PShape vector;
void setup() {
 vector = loadShape(“vector.svg”);
}
void draw() {
 shape(vector, 0, 0);
}
```

Load and Display OBJs

```
PShape object;

void setup() {

  object = loadShape(“object.obj”);
}

void draw() {

  shape(object, 0, 0);

}
```

We’ll come back to 3D geometry soon, but let’s focus on 2D examples for now…

Creating PShapes

- Same process of creating a Shape but we place the Shape into a **PShape** object!

```
PShape aShape = createShape();

aShape.beginShape();

//Set fill and stroke information here

aShape.vertex(10, 0);

aShape.vertex(100, 30);

aShape.endShape();
```

PShape Groups

- Multiple PShapes can be grouped together to create more complex shapes

```
  PShape person = createShape(GROUP);

  PShape head = createShape(ELLIPSE, 25, 25,
50, 50);

  PShape body = createShape(RECT, 0, 50, 50,
100);

  person.addChild(head);

  person.addChild(body);

  shape(person);
```

Hands-on: Using PShapes

- Today’s activities:

  1. Create a free-hand Shape using vertex points
  2. Create Shape using the TRIANGLE_STRIP parameter
  3. Create a Shape using `curveVertex`
  4. Place your free-hand Shape into a `PShape`
  5. Create a group of `PShapes` that includes at least one ellipse and one rectangle