Curves & Interactivity Lesson Plan

Jackson Eshbaugh

2026

Learning Goals

By the end of this lesson, students will be able to:

- Explain how control points influence the shape of a curve.
- Describe where a curve exists relative to its control points.
- Use `curve()` and `curveVertex()` to draw smooth curves.
- Explain why `setup()` and `draw()` are required for interactivity.
- Create interactive sketches using mouse input.

1 Curves in Processing

1.1 The `curve()` Function

```
curve(cpx1, cpy1, x1, y1, x2, y2, cpx2, cpy2);
```

1.2 Conceptual Model

- **Control points** `(cpx1, cpy1)` and `(cpx2, cpy2)` pull the curve in a direction.
- Control points tell the curve *how to bend*, not where it must go.
- The curve is *pulled* by the control points; think of this as blending directions rather than snapping to points.
- The curve only exists between `(x1, y1)` and `(x2, y2)`.
- Control points influence the shape outside that region but are not part of the drawn curve.

1.3 Multiple Curves with `curveVertex()`

- When using multiple `curveVertex()` calls, the curve exists between each pair of consecutive vertices.
- Neighboring vertices act as control points for the segment being drawn.

2 Control Point Demonstrations

2.1 Example 1: Static Comparison

- Show two curves with the same endpoints.
- Change only the control point locations.
- Emphasize how dramatically the curve changes even though the endpoints remain fixed.

2.2 Example 2: Interactive Control Points

- Start with both control points set to `mouseX` and `mouseY`.
- Observe how the curve responds in real time.
- Fix one control point at a static location.
- Vary only the remaining control point to isolate its effect.

3 Interactivity in Processing

3.1 The Two Key Functions

- `setup()` — called once at the beginning; prepares the program.
- `draw()` — called once per frame; runs in a loop.

3.2 Why This Enables Interactivity

Because `draw()` runs repeatedly, values can change over time, allowing the program to respond to input.

3.3 Mouse Variables

- `mouseX`, `mouseY`: current mouse position.
- `pmouseX`, `pmouseY`: previous frame's mouse position.

4 Live Coding Examples

4.1 Moving Ball

- Demonstrate a ball that follows the mouse.
- Emphasize calling `background()` at the start of `draw()`.
- Explain that this clears previous frames to prevent visual trails.

4.2 Drawing with Mouse Motion

- Draw lines between `mouseX, mouseY` and `pmouseX, pmouseY`.
- Show how this creates a paint- or drawing-like effect.
- Discuss why previous mouse position is essential for smooth strokes.

Wrap-Up

Reinforce the idea that curves are shaped by direction rather than fixed points, and that interactivity emerges from repetition over time. Encourage students to experiment with control points and mouse input to develop intuition.