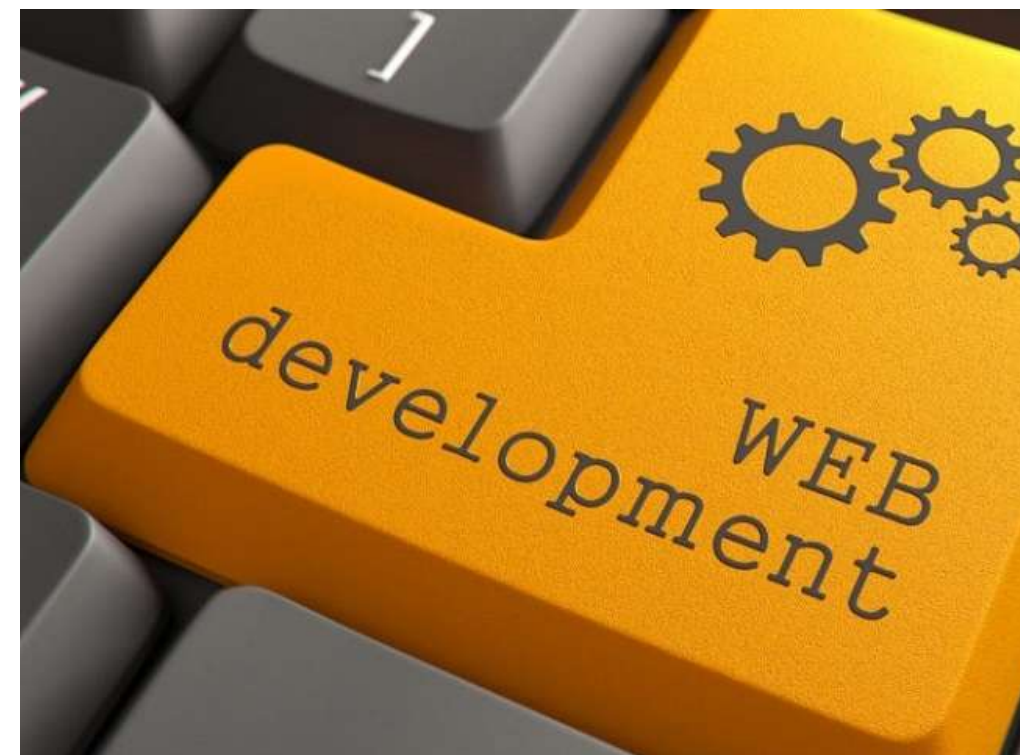




CSCI 165

Introduction to the Internet and the World Wide Web

Lec 3: Graphics IV



Jetic Gū

Overview

- Focus: Web Development
- Architecture: Internet
- Core Ideas:
 1. Paths
 2. Animation

Paths in Raphaël

Lines and Curves

Paths

- Sometimes in your SVG, you want custom shapes, like an apple logo
- Paths:
 - Straight lines
 - Curved lines
 - Hint: you will need use coordinates!



Paths: lines

- Step 1: create a canvas in your javascript, alongside the HTML file

- `canvas = Raphael('shapes', 200, 200);`

- Step 2: draw a single line with path

```
p = canvas.path('M50,10 L50,150');  
p.attr({  
  'stroke-width': '4',  
  'stroke': 'red'  
})
```

- This draws a red line from coordinate (50, 10) to (50, 150)
- M: move to the given coordinate
- L: draw a straight line to the given coordinates

Paths: lines

- You can also draw shapes in a single path

```
rect = canvas.path('M50,10 L50,150 L75,150 L75,10 L50,10');  
rect.attr({  
  'stroke-width': '4',  
  'stroke': 'red'  
})
```

- This draws a red line from coordinate (50, 10) to (50, 150) to (75,150) to (75,10) to (50,10)
- You can also replace the last `L50, 10` with `Z`, which will make it go back to the first coordinate:

```
rect = canvas.path('M50,10 L50,150 L75,150 L75,10 Z');
```

Curves

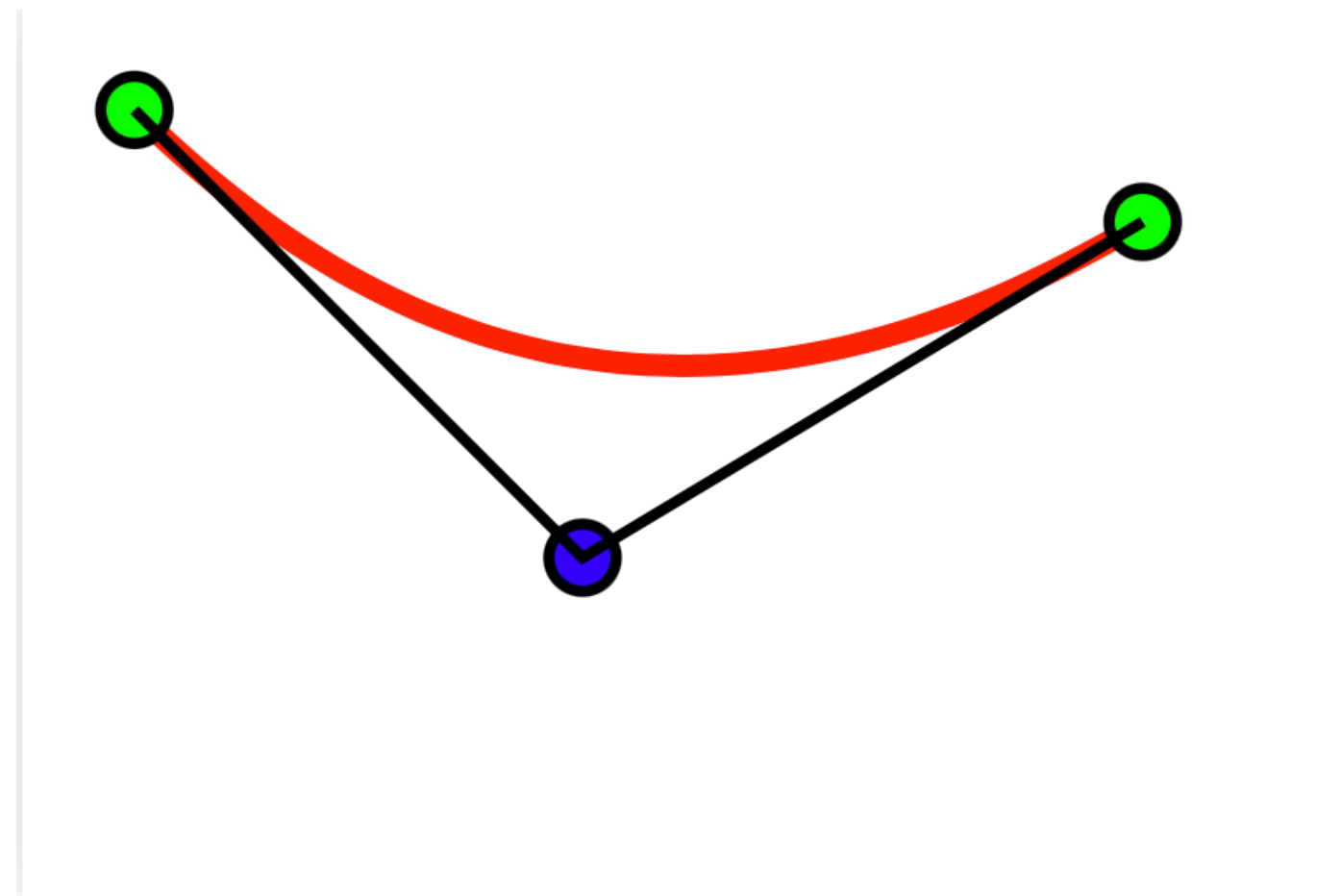
- Curves are a little bit different than straight lines
- Straight line uses L, curved lines uses Q

```
p = canvas.path('M50,10 Q50,150 100,150');
```
- With Q, you need to specify 3 points
 - the starting coordinate (this case 50,10),
 - the middle point (a point that the curve must go through, here 50,150), and
 - the ending coordinate, in this case 100,150
- You can keep drawing after this of course

Curves

- Demo:

```
curve2 = canvas.path('M10,10 Q50,50 100,20');  
curve2.attr({'stroke-width': '2', 'stroke': '#f00'});  
p1 = canvas.circle(10, 10, 3).attr({'fill': '#0f0'});  
p2 = canvas.circle(50, 50, 3).attr({'fill': '#00f'});  
p3 = canvas.circle(100, 20, 3).attr({'fill': '#0f0'});  
l1 = canvas.path('M10,10 L50,50 L100,20');
```



Animation in Raphaël

Animation by changing attributes

- With Raphaël, we can set element appearance with `.attr()`
- We can set animation using `.animate()`, in the same format as jquery animation

- Here's an example:

```
rect1 = canvas.rect(10, 10, 50, 80).attr(
    'stroke-width': '4', 'stroke': 'red')
new_size = {
    'width': '80',
    'height': '50'
}
rect1.animate(new_size, 1000) // 1000 here is 1sec
```

Rotation

- Here's an example:

```
rect2 = canvas.rect(10, 10, 50, 80).attr(
    'stroke-width': '4', 'stroke': 'red')
original = {
    'transform': 'r0'
}
turned = {
    'transform': 'r360'
}
rect2.attr(original);
rect2.animate(turned, 2000);
```

- In `transform`, `r` is for rotation, `t` is for translation (move), `s` is for scaling.

Animation Options

- There are two more options we can give `.animate()`:
 - An “easing” type: should the movement be linear, accelerating, bouncy, etc. Default is `'linear'`.
 - A callback function: something to do when the animation is finished.

```
rect3 = canvas.rect(100, 20, 50, 80);
slide = {
  'transform': 't50,50'
}
rect3.animate(slide, 1000, 'linear', recolour);
recolour = function() {
  blush = {
    'fill': '#f99'
  }
  rect3.animate(blush, 1000);
}
```

Repeating Animations

```
setup = function() {
  canvas = Raphael('container', 200, 200);
  c = canvas.circle(100, 100, 40);
  grow();
}
$(document).ready(setup)

grow = function() {
  bigger = {
    'transform': 's2'
  }
  c.animate(bigger, 1000, 'linear', shrink); // Callback function shrink
}
shrink = function() {
  smaller = {
    'transform': 's1'
  }
  c.animate(smaller, 1000, 'linear', grow); // Callback function grow
}
```