CSE 190 M Flash Sessions

Session 2

Alex Miller, Spring 2011

Loops & Conditionals

For loop

```
for (var i:int = 0; i < 20; i++) {
  // code here
}
```

While loop

```
while (x == 3) {
  // code here
}
```

If statement

```
if (x == 3) {
  // code here
} else if (x > 4) {
  // code here
} else {
  // code here
}
```

Java

Casting

```
int x = 42;
double y = (double) x;
```

String equality

```
String x = “hello”;
String y = “hello”;
if (x.equals(y)) {
   // ...
}
```

Constant

```
public static final int x = 42;
```

Flash

Casting

```
var x:int = 42;
var y:Number = x as int;
```

String equality

```
var x:String = “hello”;
var y:String = “hello”;
if (x == y) {
   // ...
}
```

Constant

```
public static const x:int = 42;
```

Java

Casting

```
int x = 42;
double y = (double) x;
```

String equality

```
String x = “hello”;
String y = “hello”;
if (x.equals(y)) {
   // ...
}
```

Constant

```
public static final int x = 42;
```

Any others?

Flash

Casting

```
var x:int = 42;
var y:Number = x as int;
```

String equality

```
var x:String = “hello”;
var y:String = “hello”;
if (x == y) {
   // ...
}
```

Constant

```
public static const x:int = 42;
```

Sprite

- Sprites are our main graphical building block.
- We can create a new empty Sprite:

```
import flash.display.Sprite;
...
var mySprite:Sprite = new Sprite();
```

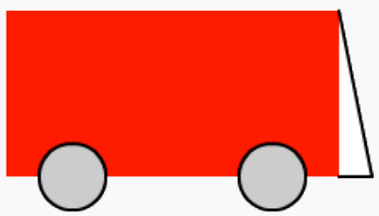
- We can write classes that extend Sprite. Useful because we get a Graphics object for each Sprite.

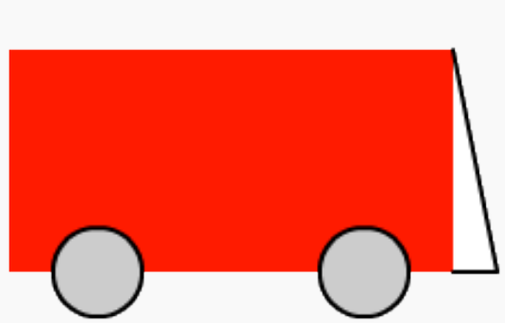
Sprite

AS	Description
`sprite.graphics;`	Graphics object used to draw on the Sprite.
`sprite.x;` `sprite.y;`	The (x, y) coordinates of the Sprite. You can change these to move the Sprite.
`sprite.width;` `sprite.height;`	Width and height of the Sprite.
`sprite.addChild(child);`	Used to add other Sprites onto the Sprite.
`sprite.stage;`	A reference to the “stage” for the Flash movie.
`sprite.visible;`	Boolean that determines whether Sprite is visible or not.
`sprite.rotation`	Angle of rotation in degrees. Default is 0.
`sprite.alpha`	Value between 0 and 1 which indicates how transparent the sprite is.

http://livedocs.adobe.com/flash/9.0/ActionScriptLangRefV3/flash/display/Sprite.html

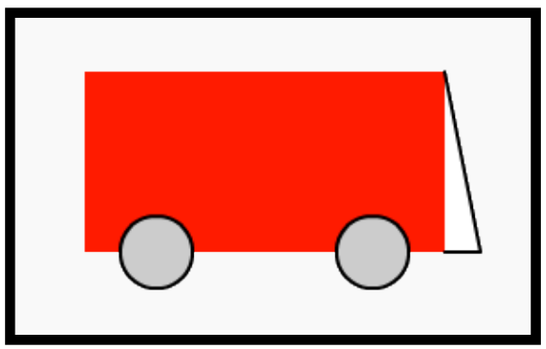
MyProgram 





MyProgram

truck



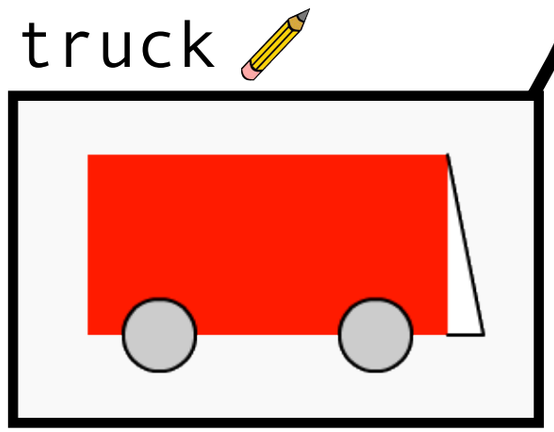
MyProgram

truck

MyProgram

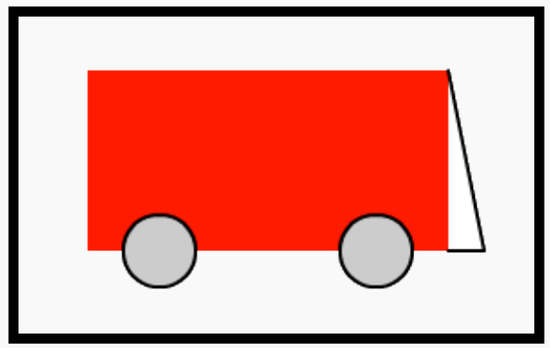
addChild(truck);

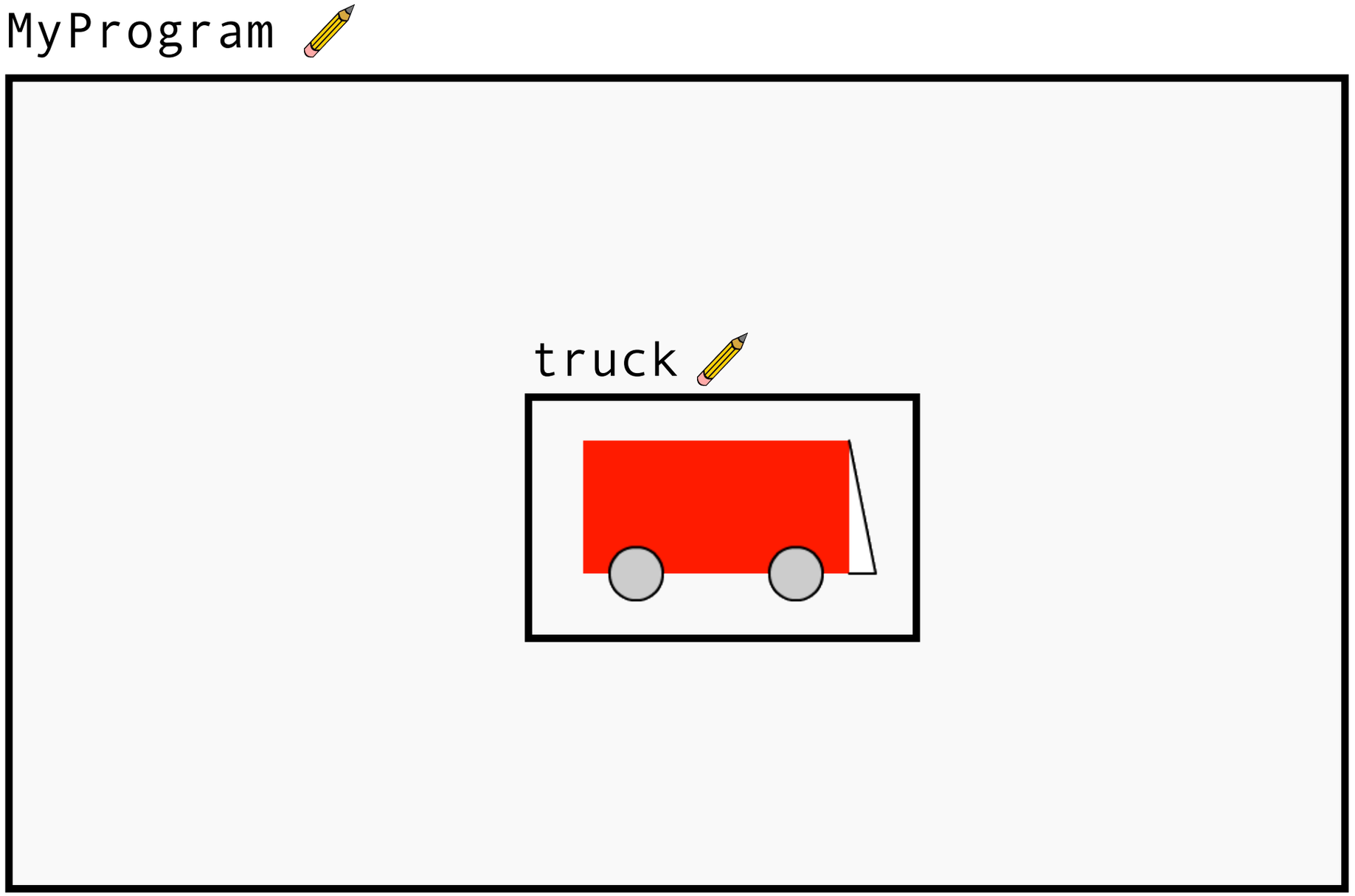
truck



MyProgram

truck





Confusing subtlety: Graphics draws shapes relative to the (x, y) position of the Sprite it’s attached to.

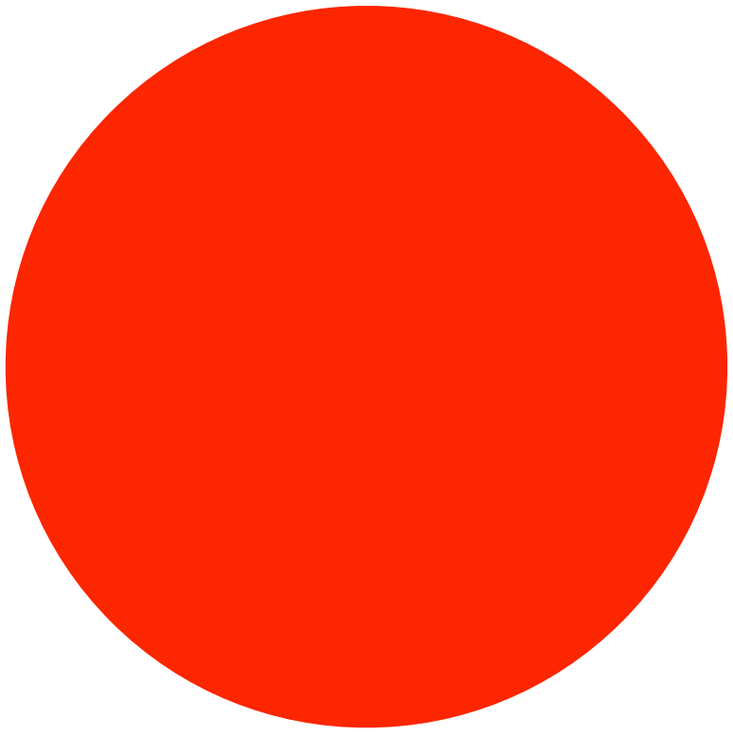
Stage

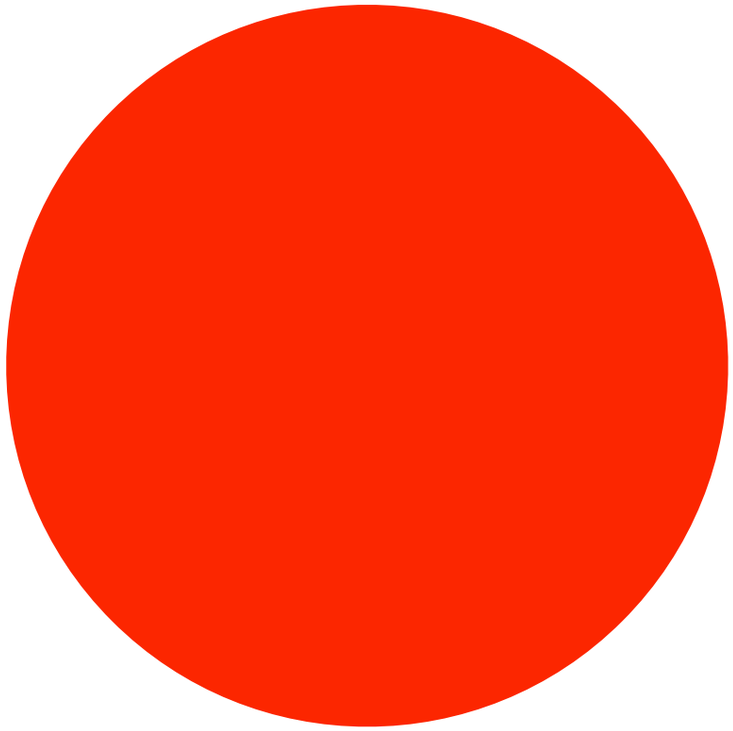
- The main container for everything, even the document class Sprite.
- All Sprites have a reference to the stage for the movie.
- Holds information about the movie:

AS	Description
`stage.stageWidth;` `stage.stageHeight;`	Overall width and height of movie.
`stage.mouseX;` `stage.mouseY;`	Mouse (x, y) coordinates.

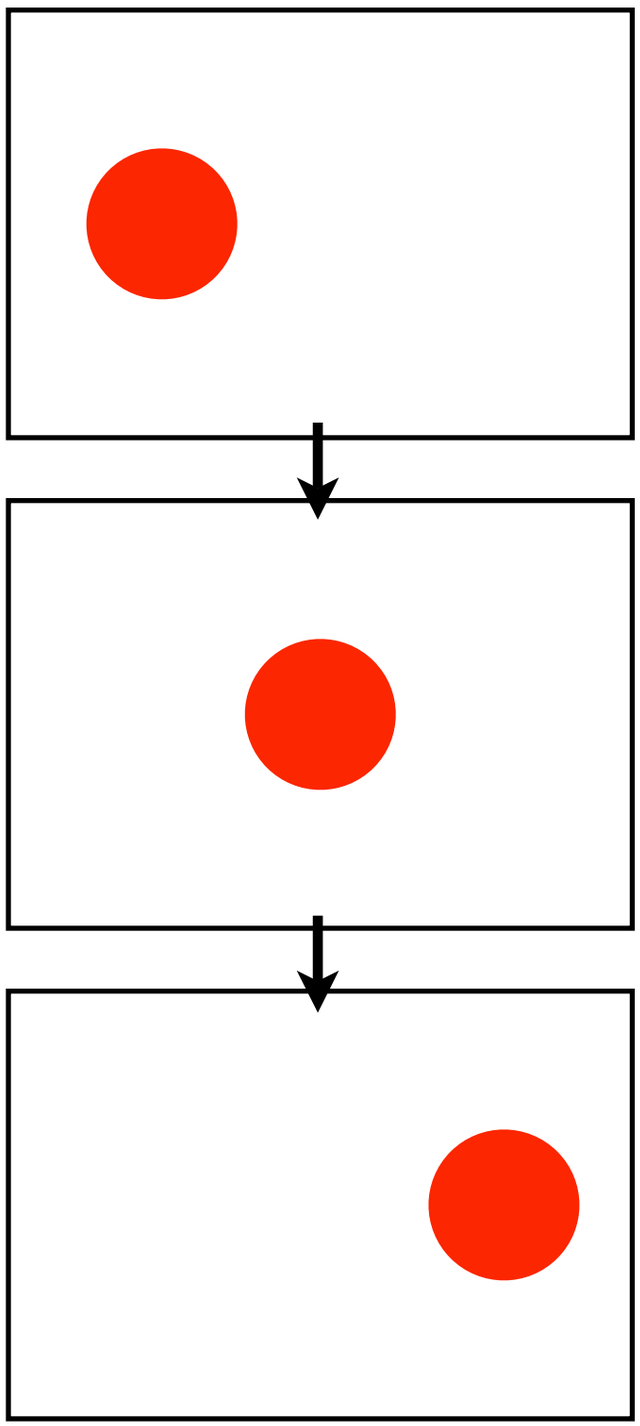
http://livedocs.adobe.com/flash/9.0/ActionScriptLangRefV3/flash/display/Stage.html

Animation





Animation



- An animation consists of a series of frames.
- Each frame is displayed on the screen for a short time (~50 times per second).
- Creates the illusion of movement.

Animation

“Enter Frame” Event

```
package {
    import flash.display.Sprite;
    import flash.events.Event;

    public class MyAnimatedProgram extends Sprite {
        public function MyAnimatedProgram():void {
            // setup code
            addEventListener(Event.ENTER_FRAME, onEnterFrame);
        }

        private function onEnterFrame(e:Event):void {
            // this code runs multiple times a second
        }
    }
}
```

Math

AS	Description
`Math.abs(value);`	Returns absolute value of passed in value.

`Math.sin(value);`
`Math.cos(value);`
`Math.tan(value);`
	Returns sin/cos/tan of value (radians).

`Math.atan2(value);` `Math.atan2(y, x);`	Arctangent functions.
`Math.log(value);`	Returns the natural log of value.
`Math.max(value1, value2);` `Math.min(value1, value2);`	Returns the max/min of the two values.
`Math.pow(value1, value2);` `Math.sqrt(value);`	Power and square root functions.
`Math.round(value);`	Returns rounded value.
`Math.random();`	Returns a pseudorandom number between 0 and 1.
`Math.PI, Math.E`	Constants for π and *e*.

http://livedocs.adobe.com/flash/9.0/ActionScriptLangRefV3/Math.html

Mouse interaction

- We have access to the current (x, y) position of the mouse. We can set a Sprite's position to those coordinates to make it follow the mouse.

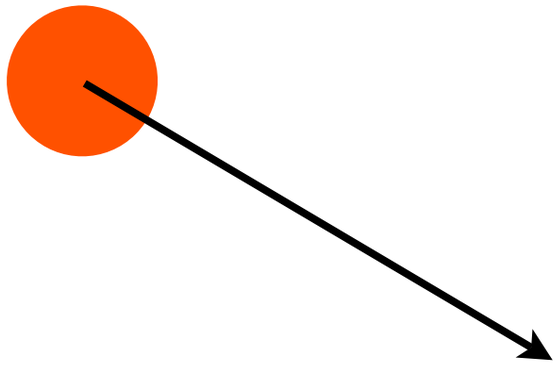
- This interaction doesn't have a lot of "pop" to it. Can we make the motion "springy" or "bouncy"?

- Instead of snapping the Sprite to the (x, y) position of the mouse every frame, we can apply a force to the Sprite in the direction of the mouse.

- To give the motion a springy effect, the force we apply every frame can be proportional to the distance from the Sprite to the mouse. A large distance means a larger force is applied that frame; a smaller distance means a smaller force.

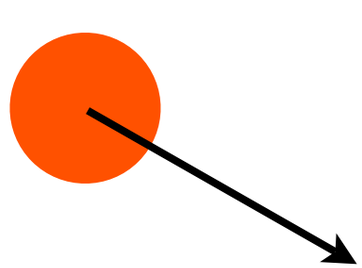
Mouse interaction

Springy Mouse Follow



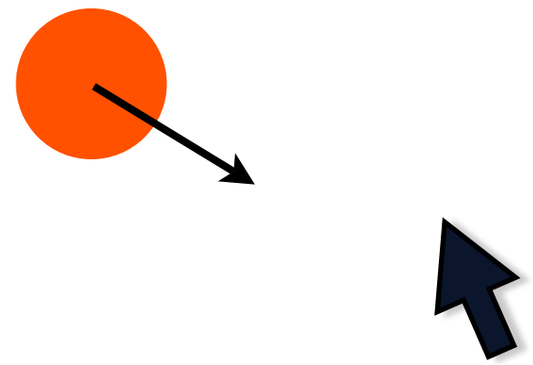
Mouse interaction

Springy Mouse Follow



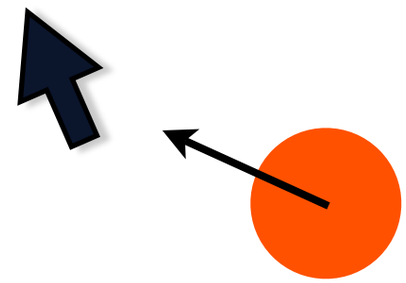
Mouse interaction

Springy Mouse Follow



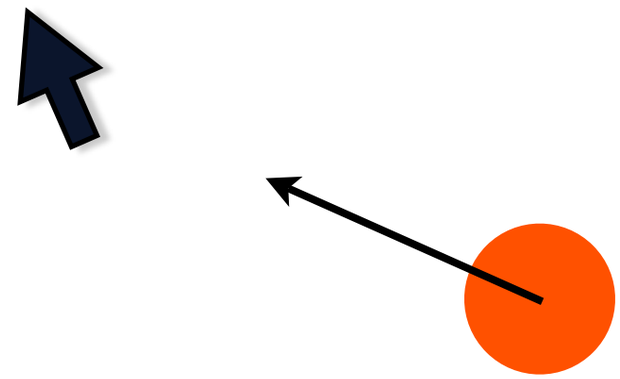
Mouse interaction

Springy Mouse Follow



Mouse interaction

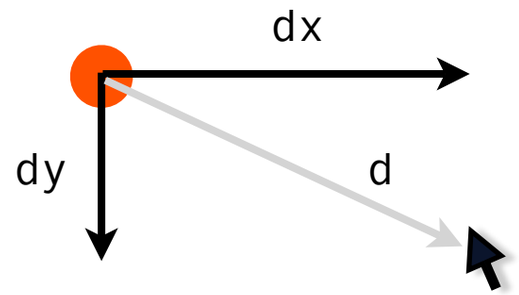
Springy Mouse Follow



Mouse interaction

Springy Mouse Follow

1. Calculate the x and y components of the displacement vector from the Sprite to the mouse.



2. Calculate the magnitude and angle of the displacement vector from `dx` and `dy`.

3. Calculate a force value based on the displacement magnitude.

4. Split the force into x and y components and add them to the x and y velocity components of the Sprite.

You may remember from physics class that F = ma. But we don't really care about mass, so to simplify things we're just doing F = a.

Next week!

- Mouse clicks, key presses
- Displaying text to the screen
- Simple game programming