html5 canvas cheat sheet

Initial Setup

```
const canvas = document.querySelector('canvas') // Grab canvas from DOM
const c = canvas.getContext('2d') // Get context to access 2D canvas functions

canvas.width = window.innerWidth // Set canvas' width to full width of window
canvas.height = window.innerHeight // Set canvas' height to full height of window
```

Drawing Basic Shapes

Rectangles - Filled

```
c.fillRect(x, y, width, height)
```

Rectangles - Stroked

```
c.strokeRect(x, y, width, height)
```

Circles & Arcs

```
c.arc(x, y, radius, startAngle, endAngle, drawClockwise)
```

JavaScript Object Blueprints

Vanilla

```
function Object(x, y, radius, color) {
    this.x = x
    this.y = y
    this.radius = radius
    this.color = color
}
```

Classes with ES6

```
class Object {
    constructor(x, y, radius, color) {
        this.x = x
        this.y = y
        this.radius = radius
        this.color = color
    }
}
```

Common Methods / Prototypes

```
Object.prototype.draw = function() {
    /* Draw canvas shapes here */
}
```

```
Object.prototype.update = function() {
    this.draw()
    /* Update object properties here */
}
```

Creating / Instantiating Objects

Singular Object

```
// Arguments should be replaced by actual values
const object = new Object(x, y, radius, color)
```

html5 canvas cheat sheet

Creating / Instantiating Objects Continued...

Multiple Objects

```
let objectArray = [] // Create holder to store multiple objects
for (let i = 0; i < 800; i++) {
    const x = Math.random() * canvas.width
    const y = Math.random() * canvas.height
    const radius = Math.random() * 5
    const color = 'blue'
    objectArray.push(new Object(x, y, radius, color)) // Store objects in holder array
}
```

Animating Objects

```
function animate() {
    requestAnimationFrame(animate) // Create an animation loop
    c.clearRect(0, 0, canvas.width, canvas.height) // Erase whole canvas

    // Animate singular object
    object.update()

    // Animate multiple objects
    objects.forEach(object => {
        object.update()
    })
}

animate() // Call the function to activate animation
```

Events

Mouse Move

```
// Object to store mouse coordinates
const mouse = {
    x: undefined,
    y: undefined
}

// Set mouse position relative to window
addEventListener('mousemove', event => {
    mouse.x = event.clientX
    mouse.y = event.clientY
})
```

Browser Resize

```
// Set canvas to size of window
addEventListener('resize', event => {
    canvas.x = window.innerWidth
    canvas.y = window.innerHeight
})
```

Other Common Event Types

```
'mouseenter'    'mouseup'    'touchmove'
'mouseleave'    'keydown'    'touchenter'
'mousedown'  'keyup'         'touchleave'
```

html5 canvas cheat sheet

Full Example

```
const canvas = document.querySelector('canvas')
const c = canvas.getContext('2d')

canvas.width = window.innerWidth
canvas.height = window.innerHeight

const mouse = {
    x: undefined,
    y: undefined
}

// Event Listeners
addEventListener('mousemove', event => {
    mouse.x = event.clientX
    mouse.y = event.clientY
})

addEventListener('resize', () => {
    canvas.width = innerWidth
    canvas.height = innerHeight

    init()
})

// Objects
function Circle(x, y, radius, color) {
    this.x = x
    this.y = y
    this.radius = radius
    this.color = color
    this.velocity = {
        x: Math.random() - 0.5, // Random x value from -0.5 to 0.5
        y: Math.random() - 0.5 // Random y value from -0.5 to 0.5
    }
}

Circle.prototype.draw = function() {
    c.beginPath()
    c.arc(this.x, this.y, this.radius, 0, Math.PI * 2, false)
    c.fillStyle = this.color
    c.fill()
    c.closePath()
}

/* Continued on next page... */
```

html5 canvas cheat sheet

Full Example Continued...

```
Object.prototype.update = function() {
    this.draw()

    this.x += this.velocity.x // Move x coordinate
    this.y += this.velocity.y // Move y coordinate
}

// Implementation
let circles
function init() {
    for (let i = 0; i < 800; i++) {
        const x = Math.random() * canvas.width
        const y = Math.random() * canvas.height
        const radius = Math.random() * 5
        const color = 'blue'
        circles.push(new Circle(x, y, radius, color))
    }
}

// Animation Loop
function animate() {
    requestAnimationFrame(animate) // Create an animation loop
    c.clearRect(0, 0, canvas.width, canvas.height) // Erase whole canvas

    circles.forEach(circle => {
        circle.update()
    })
}

init()
animate()
```