Section A
Unit #9
Classes

Content

Section A
Unit #9 Classes

Unit #9 Classes

An introduction to classes as well as objects. This is an important topic as many programming languages use classes. In this unit we will have a break from circles and use rectangles as cars. This continues to develop the concept of object orientated programming.

This is where it starts to come together. So be encouraged, if you have come this far and not lost the will to live you have done well, also this is the main place we have been heading to.

Object Orientated Programming

This explores Object Orientated Programming (OOP). We are still using p5.js and using much of what you have learned already. This is another approach to coding that is very powerful and popular. You will find most coding languages use classes and objects in some form or other.

It isn't harder than the previous two parts but you will have to think through the code to understand the process. There are fewer challenges for you and the expectation is that you will use these examples in your own projects. This third part will significantly improve your coding skills and understanding.

My advice is to work through each sketch. Alter, adapt or just play with the code to get it to do something different. That way you will help to embed those new skills.

Below is a summary of the basic structure of the classes used in this next unit. The functions display() and movement() are words we have created. Using constructor() is normal practise.

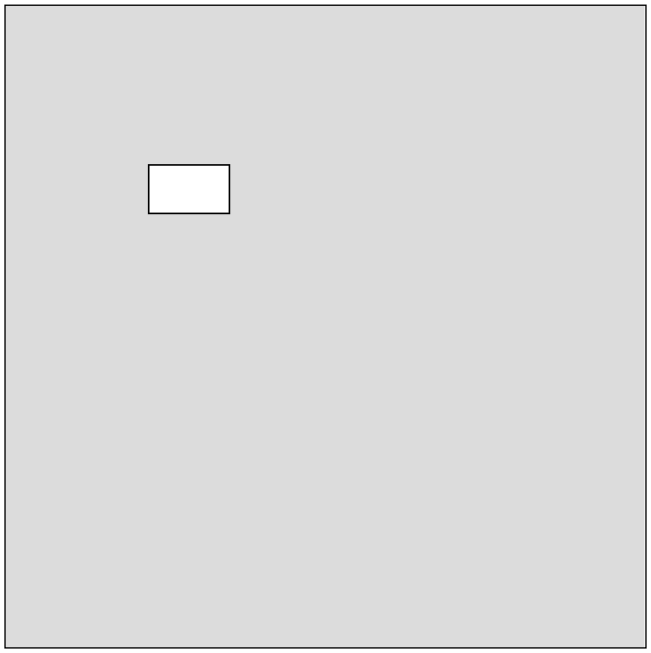
```
class xxxx
  constructor()
  {

  }

  display()
  {

  }

  movement()
  {

  }
```

Sketch A9.1 a single car

The simplest Sketch Ato move a car (starting point)

```
let x = 0

function setup()
{
  createCanvas(400, 400)
}

function draw()
{
  background(220)
  rect(x, 100, 50, 30)
  x += 1
  if (x > width)
  {
    x = 0
  }
}
```

The car is moving across the canvas



Notes

This car (rectangle) moves across the screen and then resets once it reaches the edge. There is nothing new here. It sets the scene for what is come next. Familiarise yourself with this Sketch Ayou are going to use this as the foundation for what is to come.

Challenge

Change its colour, speed or shape

Sketch A9.2 function single car

Moving the car using functions

```
let x = 0

function setup()
{
  createCanvas(440, 400)
}

function draw()
{
  display()
  movement()
}

function display()
{
  background(220)
  rect(x, 100, 50, 30)
}

function movement()
{
  x += 1
  if (x > width)
  {
    x = 0
  }
}
```

Notes

Next we add movement() and display() functions for the car. The draw() function calls those functions. This creates the same effect as the previous sketch. So far so good, it should seem familiar.

Challenge

Explore the functions to see what they do and how they work

Sketch A9.3 car as an object

Start a new sketch. Adding the car as an object

```
let car = {x: 0}

function setup()
{
  createCanvas(400, 400)
}

function draw()
{
  background(220)
  rect(car.x, 100, 50, 30)
  car.x += 1
  if (car.x > width)
  {
    car.x = 0
  }
}
```

Notes

The car is now an object with an x component

Challenge

Change the name of the object and change its shape

Sketch A9.4 alternative car object

For practise start a new sketch. A variation of the car object

```
let car = {x: 0, y: 100}

function setup()
{
  createCanvas(400, 400)
}

function draw()
{
  background(220)
  display()
  movement()
}

function display()
{
  rect(car.x, car.y, 50, 30)
}

function movement()
{
  car.x += 1
  if (car.x > width)
  {
    car.x = 0
  }
}
```

Notes

This is an alternative way of writing the Sketch Afor the object car.

Challenge

Add the length and width of the car into the object

Classes - a brief explanation

In the next two sketches, you are going to use Classes. Don't be put off with what may seem an unnecessary complication. I have kept it as simple as possible so that you can get a feel for how to create and use classes.

A class can have many objects. The Sketch A9.5 has one class Car and one object car, whereas Sketch A9.10 also has one class but two objects. Even though you only have a template for one car you can produce lots of them.

You initialise the car object in **`setup()`** with **`myCar = new Car()`**. You operate the car object in **`draw()`** by using the **`dot`** syntax **`myCar.movement()`** and **`myCar.display()`** The class is defined class Car. There are three main functions inside the class. A constructor function is defined as constructor(), the display function **`display()`** and the move function **`movement()`**. They are functions in the class and have different... functions!

The names **`constructor`**, **`display`** and **`movement`** are ones I have created. You can given them any name you like but keeping them relevant and meaningful helps you (and anyone else) remember what they are and what they are for. Although using the **`constructor`** naming convention is the industry norm.

Sketch A9.5 class with one car

This is create a car class

```
let myCar

function setup()
{
  createCanvas(400, 400)
  myCar = new Car()
}

function draw()
{
  background(220)
  myCar.display()
  myCar.movement()
}

class Car
{
  constructor()
  {
    this.colour = 255
    this.x = 0
    this.y = 100
    this.velocity = 1
  }

  display()
  {
    fill(this.colour)
    rect(this.x, this.y, 50, 30)
```

```
  }

  movement()
  {
    this.x += this.velocity
    if (this.x > width)
    {
      this.x = 0
    }
  }
}
```

Notes

A new object is initialised in setup() with myCar = new Car(). The display() and movement() functions are called from inside the draw function. The constructor() function, constructs the car, it holds data about the car.

Challenge

Try changing the names of the class, the object, constructor, display and move

The structure of a class

A basic structure and key elements for classes

```
function setup()
{
  new xxxx
}
```

```
function draw()
{
  xxxx.display()
  xxxx.movement()
}
```

```
class
  constructor()
  {
    this.xxxx
  }

  display()
  {
    xxxx
  }

  movement()
  {
    xxxx
  }
```

When you write or see other sketches with classes they may well have functions other than display or move.

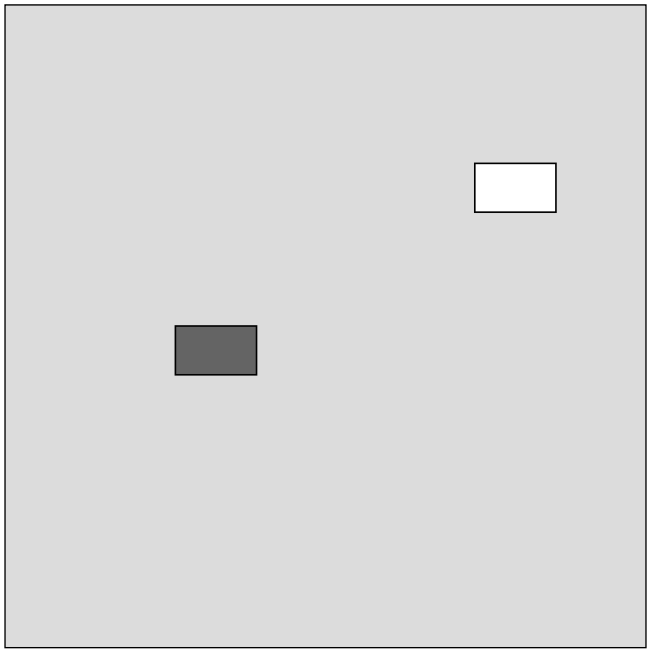
Sketch A9.6 class with two cars

I will still recommend starting with a new Sketch Abut you can see that there is a bit of repetition, so if you are careful you could copy and paste some of it with the changes. This demonstrates the beauty of classes

```
let myCar1
let myCar2

function setup()
{
  createCanvas(400, 400)
  myCar1 = new Car(255, 0, 100, 1.5)
  myCar2 = new Car(100, 0, 200, 2)
}

function draw()
{
  background(220)
  myCar1.display()
  myCar2.display()
  myCar1.movement()
  myCar2.movement()
}

class Car
{
  constructor(colour, x_position, y_position, x_velocity)
  {
    this.c = colour
    this.x = x_position
    this.y = y_position
    this.velocity = x_velocity
```

```
  }

  display()
  {
    fill(this.c)
    rect(this.x, this.y, 50, 30)
  }

  movement()
  {
    this.x += this.velocity
    if (this.x > width)
    {
      this.x = 0
    }
  }
}
```

One class two cars



Notes

There are two objects myCar1 and myCar2. The arguments go inside the brackets when the object is constructed, they is one set for the first car and another set for the second car. In this Sketch Athe constructor is defined by four parameters (colour, x_position, y_position, x_velocity).

Challenge

Make more cars all different colours

This may look a bit complicated but if you really want to learn to code these are some of the examples you need to get your head around. Object Orientated Programming increases the possibilities of being creative, it is a powerful tool that will pay dividends later on.

If you have struggled to really get to grips with it, I did at first, then you are in good company but after a while you start to see the patterns emerging and it is not so difficult, just very new.