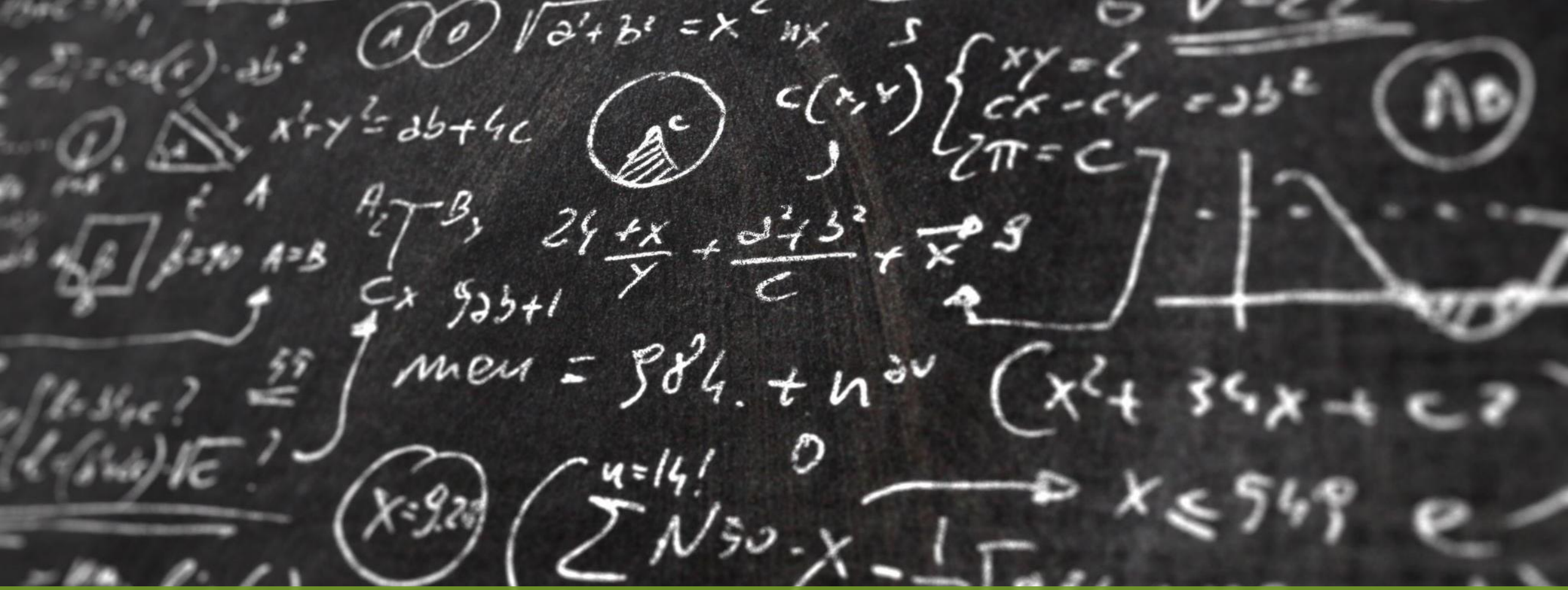


ES6 CLASSES

Prof. Andrew Sheehan

Metropolitan College
Boston University
Computer Science Department

WHAT IS A CLASS?

A class is an object that has state (properties) and behavior (methods)

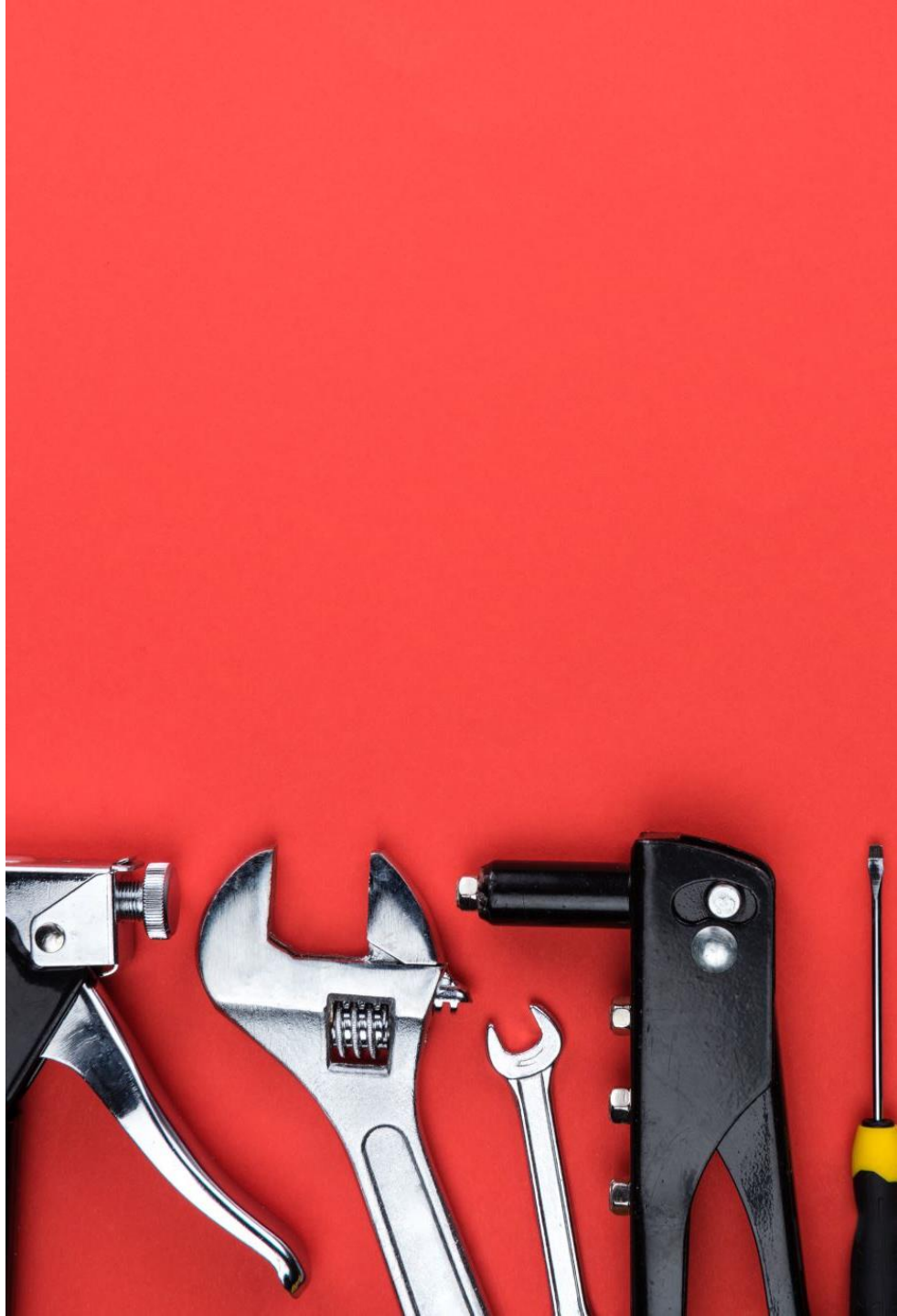
BUILT-IN TYPES

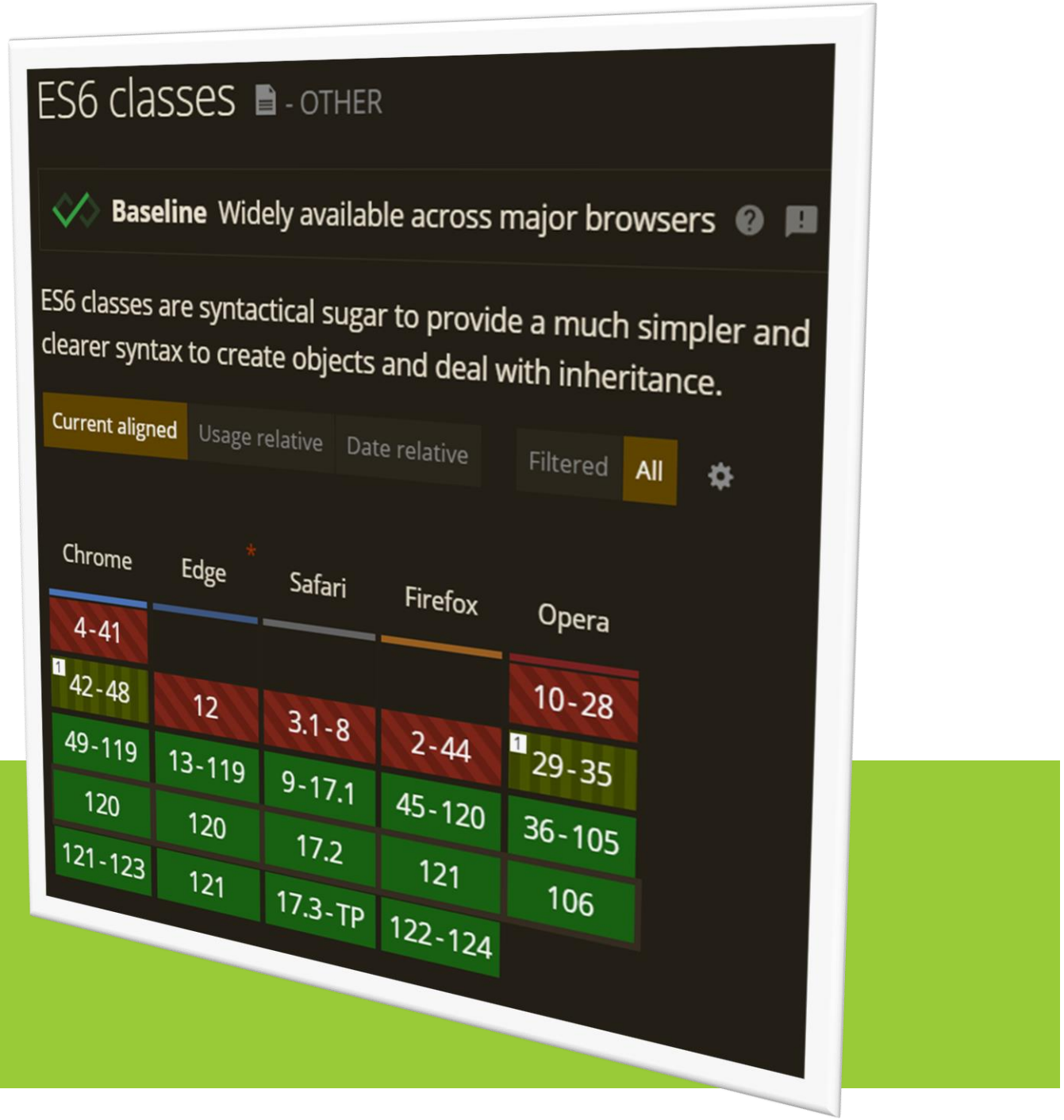
Reference types

(Array, String, Date, Set, Map..)

Primitive types

(boolean, numeric, NULL..)





You can use it with all modern browsers

```
class Rectangle {
  constructor(height, width) {
    this.height = height;
    this.width = width;
  }
}
```

CLASSIC EXAMPLE

```
const p = new Rectangle(); // ReferenceError

class Rectangle {}
```

Click to add text

DECLARE BEFORE USE

CLASS EXPRESSIONS

```
// unnamed
let Rectangle = class {
  constructor(height, width) {
    this.height = height;
    this.width = width;
  }
};
console.log(Rectangle.name);
// output: "Rectangle"

// named
let Rectangle = class Rectangle2 {
  constructor(height, width) {
    this.height = height;
    this.width = width;
  }
};
console.log(Rectangle.name);
// output: "Rectangle2"
```

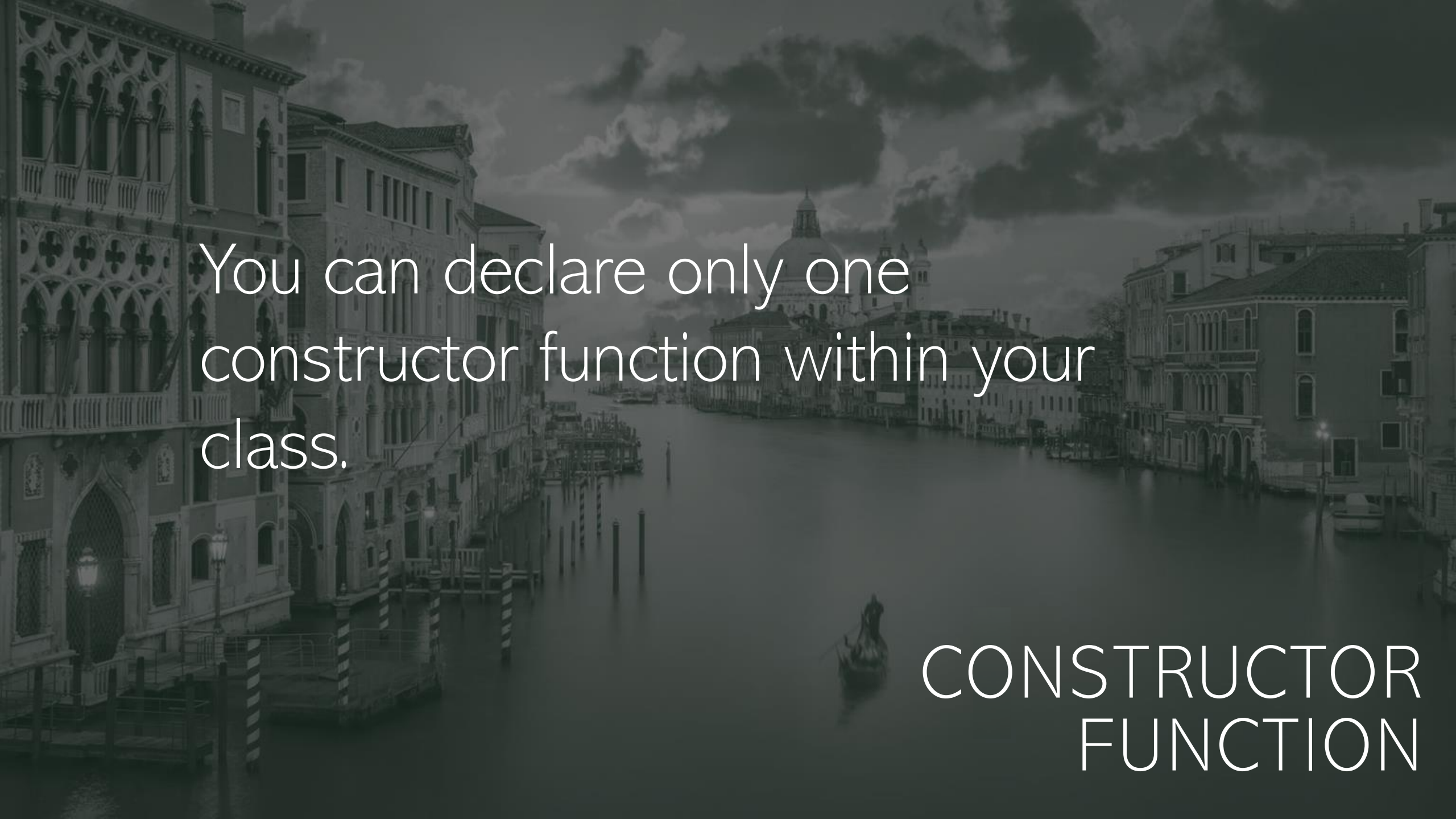
Note: Class **expressions** must be declared before they can be used

BODY OF A CLASS

```
class Rectangle {
  constructor(height, width) {
    this.height = height;
    this.width = width;
  }
  // Getter
  get area() {
    return this.calcArea();
  }
  // Method
  calcArea() {
    return this.height * this.width;
  }
}
```

Start of class body

End of class body



You can declare only one constructor function within your class.

CONSTRUCTOR FUNCTION

INSTANCE PROPERTIES

01 You can define properties **within the constructor**

02 By default, properties & methods **are public**

```
class Rectangle {
  constructor(height, width) {
    this.height = height;
    this.width = width;
  }
}
```

PRIVATE: USE THE

Private instance properties are called **"hash names"**

```
class Rectangle {
  #height = 0;
  #width;

  constructor(height, width) {
    this.#height = height;
    this.#width  = width;
  }
}
```

PRIVATE:
NO ACCESS OUTSIDE CLASS

```
class Human {
  #bloodType;
  constructor(type) {
    this.#bloodType = type;
  }
}

// in your code someplace...
let me = new Human("A-Pos");
console.info(me.#bloodType);   //error: 'undefined'
```

USING ACCESSORS WITH PRIVATE ATTRIBUTES

```
class Human {
    #bloodType;
    constructor(type) {
        this.#bloodType = type;
    }
    get typeOfBlood() {
        return this.#bloodType;
    }
}
```

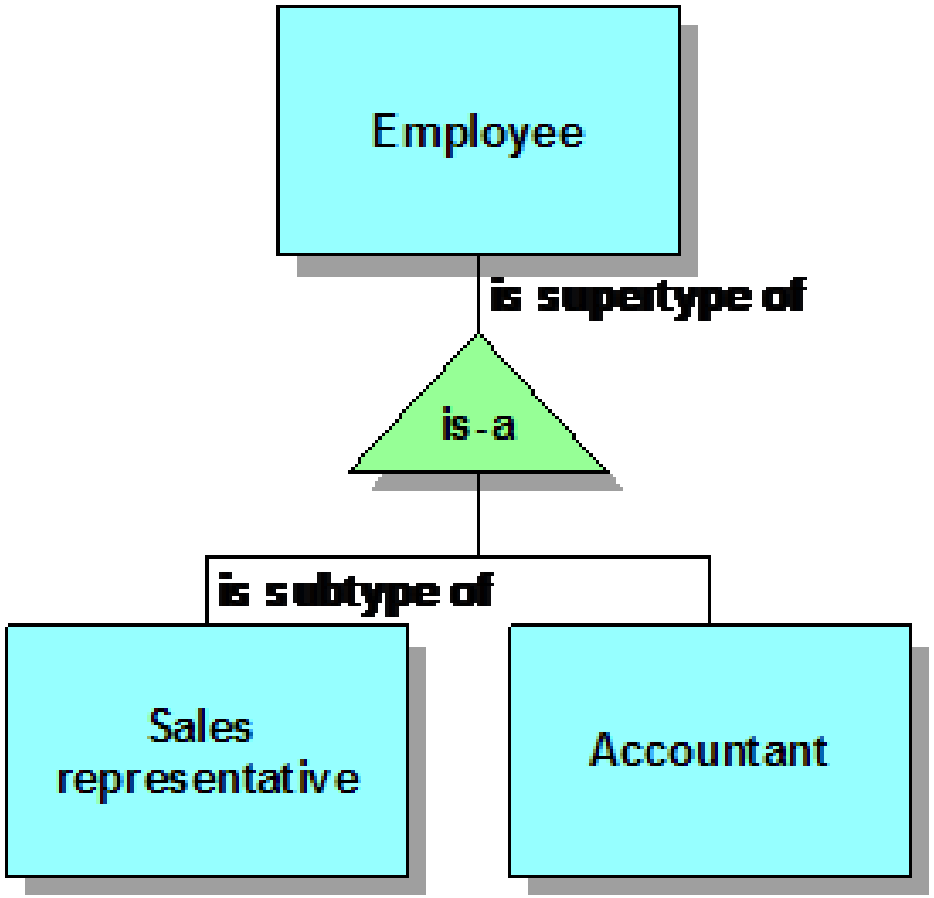
```
class Point {
  constructor(x, y) {
    this.x = x;
    this.y = y;
  }

  static displayName = "Point";
  static distance(a, b) {
    const dx = a.x - b.x;
    const dy = a.y - b.y;

    return Math.hypot(dx, dy);
  }
}
```

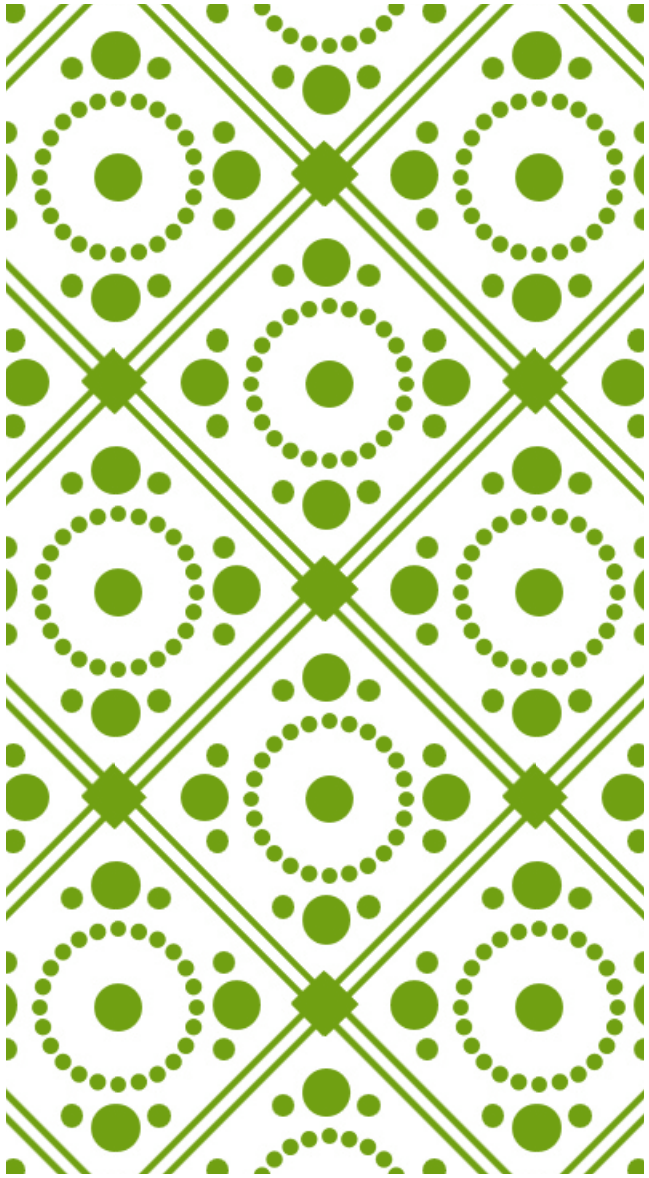
```
const p1 = new Point(5, 5);
const p2 = new Point(10, 10);
p1.displayName; // undefined
p1.distance;    // undefined
p2.displayName; // undefined
p2.distance;    // undefined

console.log(Point.displayName);      // "Point"
console.log(Point.distance(p1, p2)); // 7.0710678118654755
```

STATIC METHODS AND PROPERTIES

SUBCLASSES
GENERALIZATION

A subclass is a type of class that inherits from a base class





super()

Use super() to access your methods and attributes in your parent class.

```
class Cat {
  constructor(name) {
    this.name = name;
  }

  speak() {
    console.log(`${this.name} makes a noise.`);
  }
}
```

A PARENT CLASS

```
class Lion extends Cat {
  speak() {
    super.speak();
    console.log(`${this.name} roars.`);
  }
}

let l = new Lion('Fuzzy');
l.speak();
// Fuzzy makes a noise.
// Fuzzy roars.
```

A GENERALIZED TYPE