



getter e setter

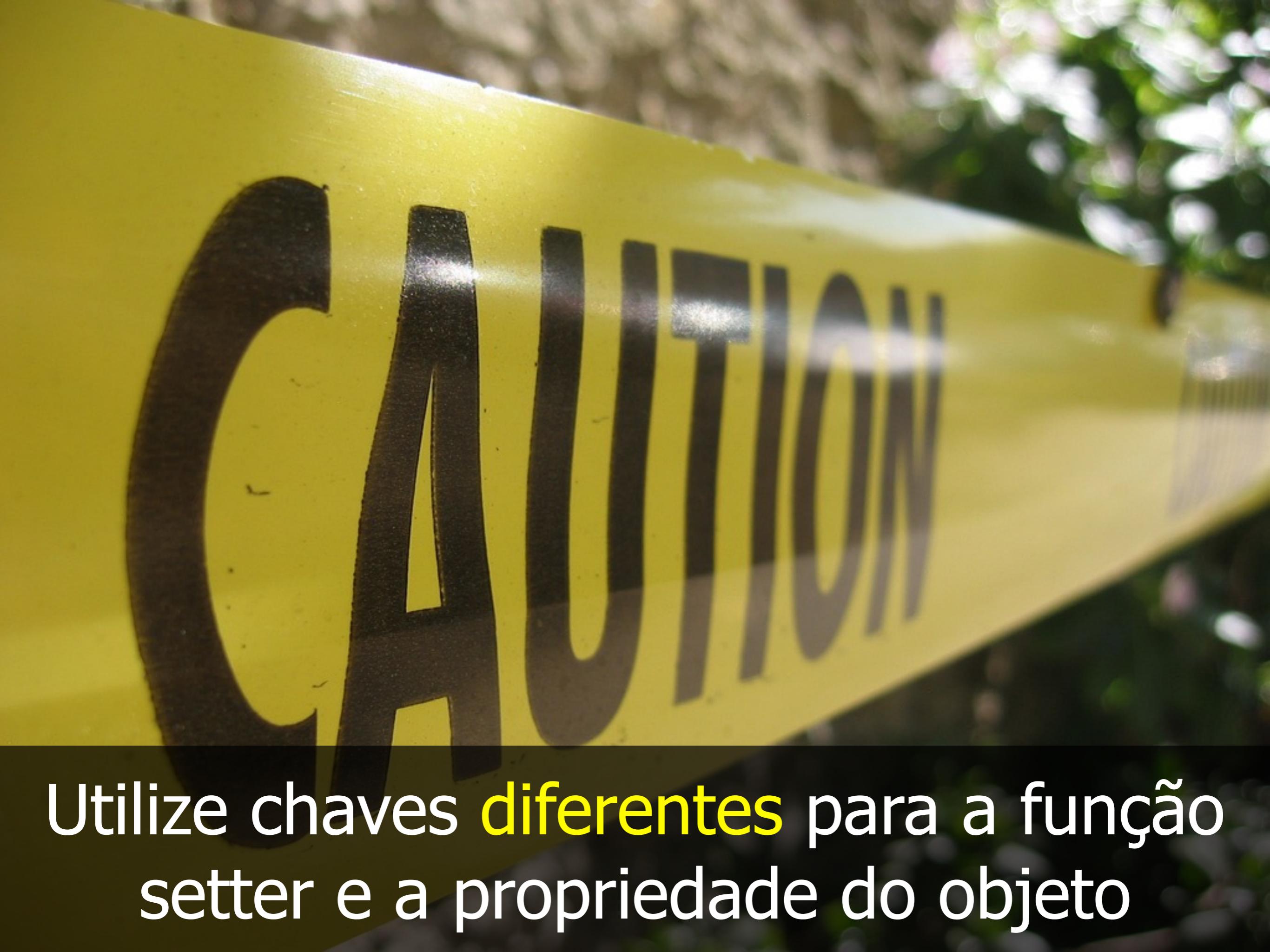
As funções do tipo **getter** e **setter** servem para interceptar o acesso as propriedades de determinado um objeto

A screenshot of a macOS desktop environment. On the left is a code editor window titled "getter_setter_1.js — javascriptmasterclass". The file contains the following JavaScript code:

```
JS getter_setter_1.js ×
1 const rectangle = {
2   x: 10,
3   y: 2,
4   get area() {
5     return this.x * this.y;
6   }
7 };
8 console.log(rectangle.area);
9
```

The code defines a constant `rectangle` which is an object with properties `x` and `y`, and a computed property `area` which returns the product of `x` and `y`. A call to `console.log` at the end prints the value of `area`.

To the right of the code editor is a terminal window titled "1: bash". The terminal shows the command `node getter_setter/getter_setter_1.js` being run, followed by the output `20`.



CAUTION

Utilize chaves diferentes para a função
setter e a propriedade do objeto

A screenshot of a macOS desktop environment. On the left is a code editor window titled "getter_setter_3.js — javascriptmasterclass". The code editor contains the following JavaScript code:

```
JS getter_setter_3.js ×
1 const rectangle = {
2     set x(x) {
3         this._x = x;
4     },
5     set y(y) {
6         this._y = y;
7     },
8     get area() {
9         return this._x * this._y;
10    }
11 };
12 rectangle.x = 10;
13 rectangle.y = 2;
14 console.log(rectangle.area);
15
```

The code defines a rectangle object with properties x and y using getters and setters, and a method area that returns the product of x and y. It then sets x to 10, y to 2, and logs the area to the console.

To the right of the code editor is a terminal window titled "TERMINAL" with the identifier "1: bash". The terminal shows the command \$ node getter_setter/getter_setter_3.js being run, followed by the output 20, which is the expected area of the rectangle.

The screenshot shows a terminal window with the following output:

```
rodrigobranas:javascriptmasterclass $ node getter_setter/getter_setter_4.js
Invalid value for x
Invalid value for y
NaN
rodrigobranas:javascriptmasterclass $
```

The code in the file 'getter_setter_4.js' is as follows:

```
1 const rectangle = {
2     set x(x) {
3         if (x > 0) {
4             this._x = x;
5         } else {
6             console.log("Invalid value for x");
7         }
8     },
9     set y(y) {
10        if (y > 0) {
11            this._y = y;
12        } else {
13            console.log("Invalid value for y");
14        }
15    },
16    get area() {
17        return this._x * this._y;
18    }
19 };
20 rectangle.x = -10;
21 rectangle.y = -2;
22 console.log(rectangle.area);
23
```



A screenshot of a terminal window titled "getter_setter_5.js — javascriptmasterclass". The terminal shows the command "node getter_setter/getter_setter_5.js" being run, followed by the output "20".

```
JS getter_setter_5.js ×
1  const rectangle = {
2      set x(x) {
3          if (x > 0) {
4              this._x = x;
5          } else {
6              console.log("Invalid value for x");
7          }
8      },
9      set y(y) {
10         if (y > 0) {
11             this._y = y;
12         } else {
13             console.log("Invalid value for y");
14         }
15     },
16     get area() {
17         return this._x * this._y;
18     }
19 };
20 rectangle.x = 10;
21 rectangle.y = 2;
22 console.log(rectangle.area);
23
```

Por meio da operação `defineProperty` da Object API, também é possível definir funções do tipo `getter` e `setter`

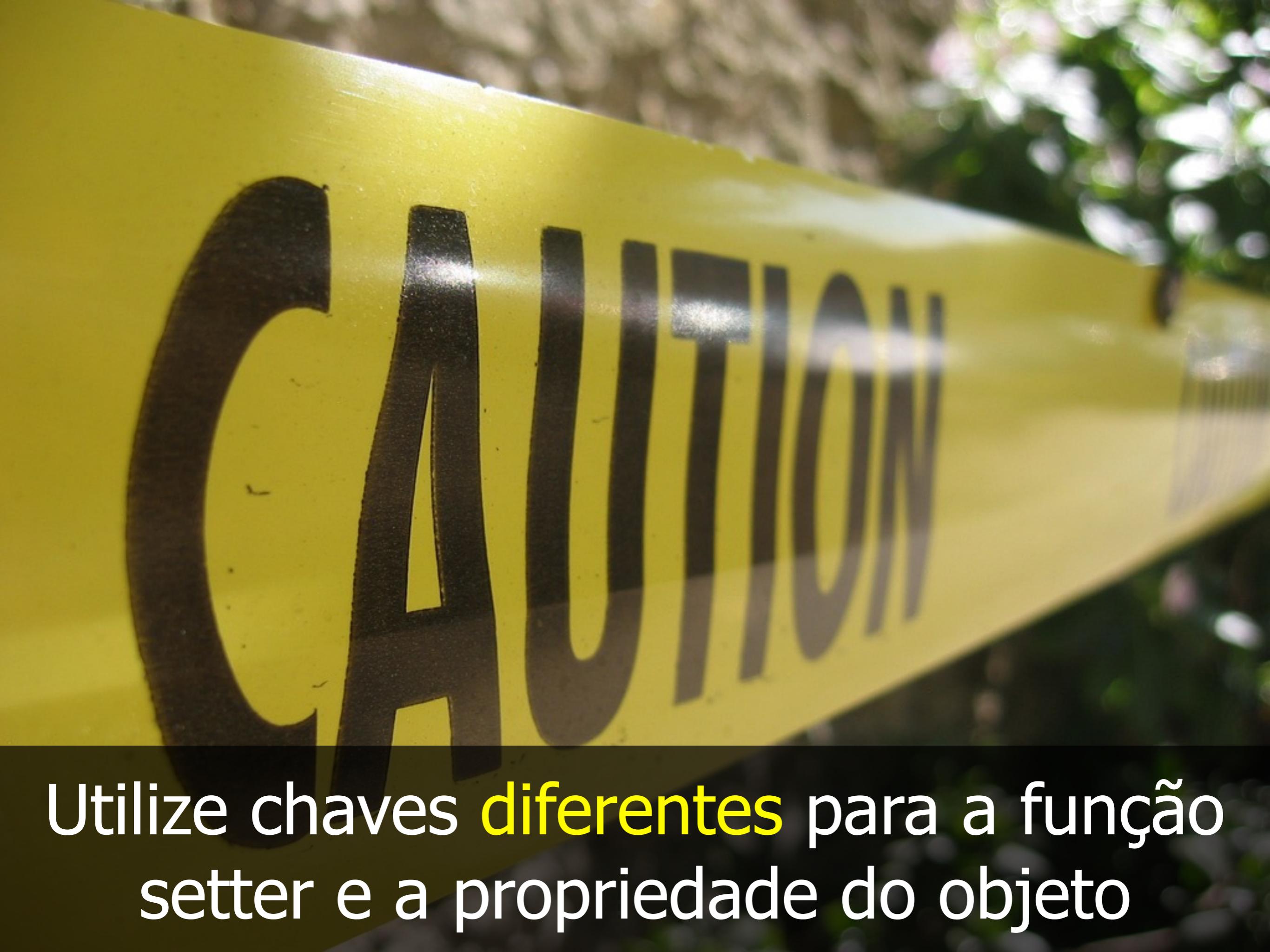
getter_setter_6.js — javascriptmasterclass

JS getter_setter_6.js x

TERMINAL ... 1: bash

```
1 const rectangle = {};
2 Object.defineProperty(rectangle, "area", {
3     get() {
4         return this.x * this.y;
5     }
6 });
7 rectangle.x = 10;
8 rectangle.y = 2;
9 console.log(rectangle.area);
10
```

rodrigobranas:javascriptmasterclass \$ node getter_setter/getter_setter_6.js
20
rodrigobranas:javascriptmasterclass \$



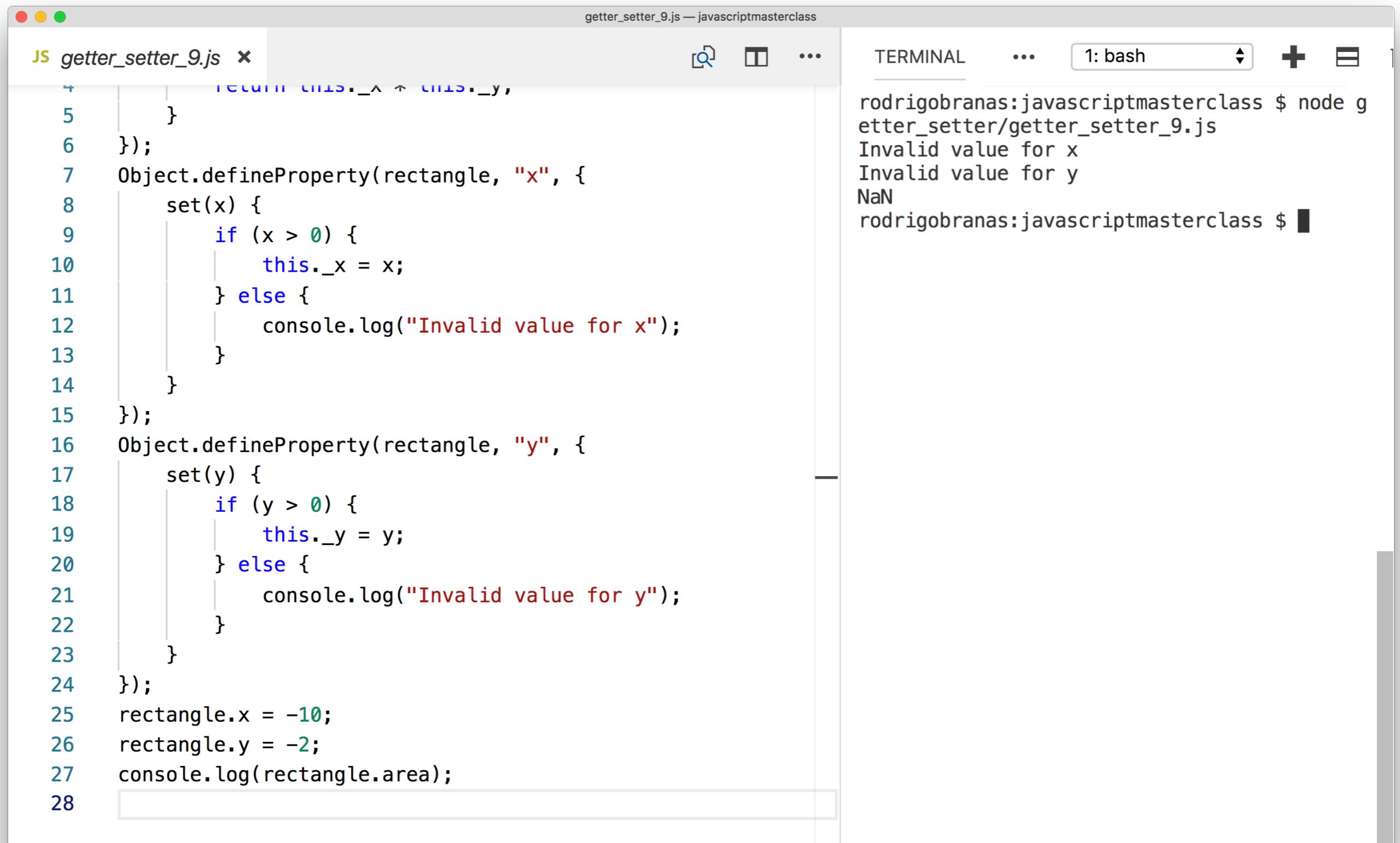
CAUTION

Utilize chaves diferentes para a função
setter e a propriedade do objeto

A screenshot of a Mac OS X desktop environment. On the left, a code editor window titled "getter_setter_8.js — javascriptmasterclass" displays a JavaScript file. The file contains code for defining a rectangle object with properties "area", "x", and "y" using Object.defineProperty, along with setter and getter functions. Lines 1 through 20 are numbered on the left. On the right, a terminal window titled "1: bash" shows the command \$ node getter_setter/getter_setter_8.js being run, followed by the output 20, which is the calculated area of the rectangle.

```
JS getter_setter_8.js ×
getter_setter_8.js — javascriptmasterclass
1 const rectangle = {};
2 Object.defineProperty(rectangle, "area", {
3     get() {
4         return this._x * this._y;
5     }
6 });
7 Object.defineProperty(rectangle, "x", {
8     set(x) {
9         this._x = x;
10    }
11 });
12 Object.defineProperty(rectangle, "y", {
13     set(y) {
14         this._y = y;
15    }
16 });
17 rectangle.x = 10;
18 rectangle.y = 2;
19 console.log(rectangle.area);
20
```

```
rodrigobranas:javascriptmasterclass $ node getter_setter/getter_setter_8.js
20
rodrigobranas:javascriptmasterclass $
```



A screenshot of a Mac OS X desktop environment showing a terminal window and a code editor. The terminal window is titled 'bash' and contains the following text:

```
rodrigobranas:javascriptmasterclass $ node getter_setter/getter_setter_9.js
Invalid value for x
Invalid value for y
NaN
rodrigobranas:javascriptmasterclass $
```

The code editor window is titled 'getter_setter_9.js' and shows the following JavaScript code:

```
JS getter_setter_9.js x
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

function Rectangle(x, y) {
    this._x = x;
    this._y = y;
}

Rectangle.prototype.area = function() {
    return this._x * this._y;
}

Object.defineProperty(Rectangle, "x", {
    set(x) {
        if (x > 0) {
            this._x = x;
        } else {
            console.log("Invalid value for x");
        }
    }
});

Object.defineProperty(Rectangle, "y", {
    set(y) {
        if (y > 0) {
            this._y = y;
        } else {
            console.log("Invalid value for y");
        }
    }
});

rectangle.x = -10;
rectangle.y = -2;
console.log(rectangle.area);
```



A screenshot of a Mac OS X desktop environment showing a terminal window and a code editor. The terminal window is titled 'bash' and contains the command 'node getter_setter/getter_setter_10.js' followed by the number '20'. The code editor window is titled 'getter_setter_10.js' and shows a JavaScript file with code for a rectangle class using getters and setters.

```
JS getter_setter_10.js x
5     }
6 });
7 Object.defineProperty(rectangle, "x", {
8     set(x) {
9         if (x > 0) {
10            this._x = x;
11        } else {
12            console.log("Invalid value for x");
13        }
14    }
15 });
16 Object.defineProperty(rectangle, "y", {
17     set(y) {
18         if (y > 0) {
19             this._y = y;
20         } else {
21             console.log("Invalid value for y");
22         }
23     }
24 });
25 rectangle.x = 10;
26 rectangle.y = 2;
27 console.log(rectangle.area);
28
```

TERMINAL ... 1: bash

```
rodrigobranas:javascriptmasterclass $ node getter_setter/getter_setter_10.js
20
rodrigobranas:javascriptmasterclass $
```