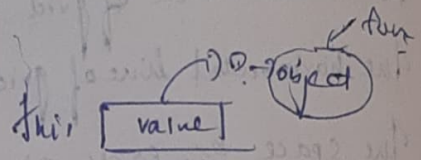
## 'this' keyword in Javascript

* this

- Keyword in JS which contains the value of the object on which particular function was called.

```
Eg: const obj = {
        name = 'foo',
        sayName() {
            console.log(this.name);
        }
    }
obj.sayName(); // prints 'foo'
```




- helpful when there are more than one object that share same prototype / class when using classes.
- Value of this inside sayName() would have been the obj.
- should be used always when accessing the members of a class inside the class.
- Is set only if invoked directly
  The value of this will be set only when the method called directly from the object.
- It does not work
  Eg: If you assign it to a different variable & call it.
- The value of this depends on whether it is [Strict mode / Normal mode]
- When accessing outside a function, the value of this will be global object.
  Window → in browsers  global → in NodeJS.

(201)

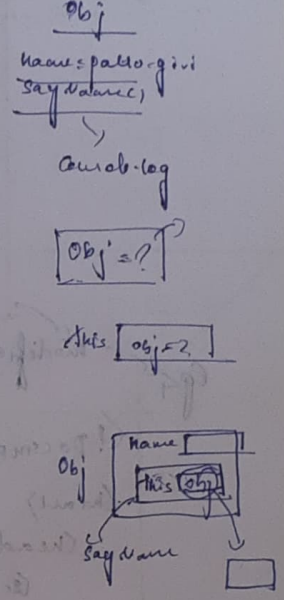
- Arrow functions dont have this.
  - this is applicable only inside normal functions.
- In arrow functions there is no special this value. It takes whatever is available outside of the arrow function.

In some cases we may want to call function with specific this value.
In such cases we can use (bind, call, apply)

Eg1)

```
<!DocType html>
<html>
  <head>
    <script>
      const obj = {
        name : 'patto-giri';
        sayName() {
          console.log(this.name);
        }
      }
      obj.sayName(); //prints 'patto-giri'
      // Here obj constant is the object
    </script>
  </head>
</html>
```

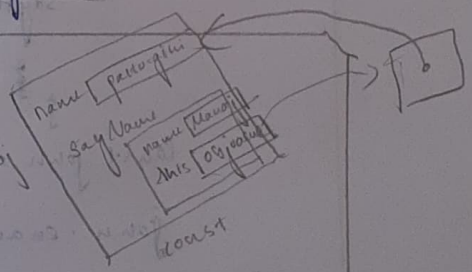
o/p : Patto-giri




Eg2)

```
<!DocType html>
<html>
  <head>
    <script>
      const obj = {
        name : 'patto-giri';
        sayName() {
          name : 'Manoj' ; //It will not print the Manoj
          console.log(this.name);
        }
      }
      obj.sayName();
      // Here obj constant is the object
    </script>
  </head>
</html>
```

o/p : patto-giri



Eg3) // $name a double instance

```
<!DOCTYPE html>
<html>
  <head>
    <script>
      Class Human{
        sayHi(){
          console.log(`${this.ename} says Hi`);
        }
      }
      const john = new Human();
      john.ename = 'John';
      john.sayHi(); //John says Hi
    </script>
  </head>
</html>
```

Class Human — sayHi

Eg4) ← modified program

```
<!DOCTYPE html>
<html>
  <head>
    <script>
      Class Human{
        sayHi(){
          console.log(this.ename + 'says Hi');
        }
      }
      const john = new Human();
      john.ename = 'John';
      john.sayHi(); //John says Hi
    </script>
  </head>
</html>
```

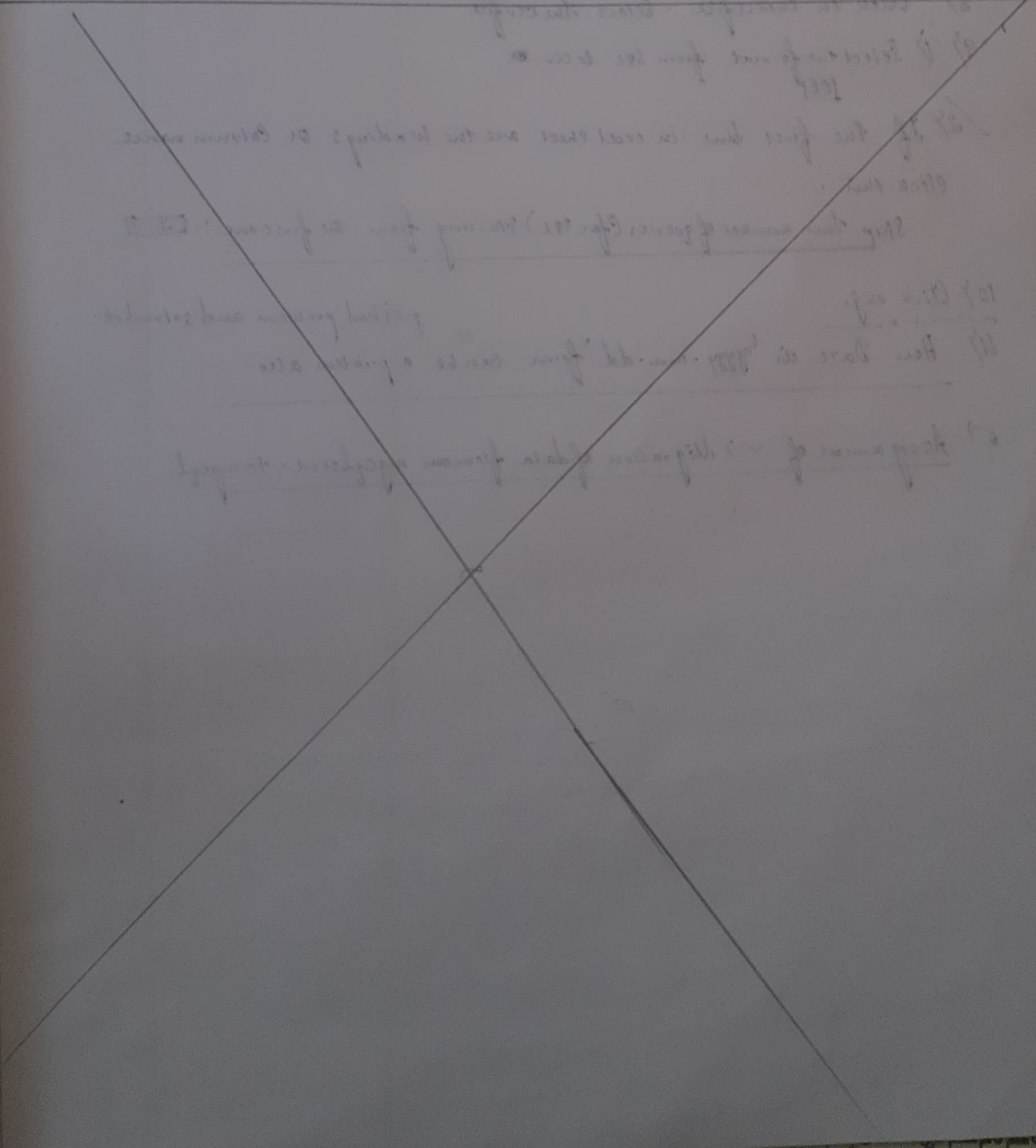
Eg5)

```
<!DOCTYPE html>
<html>
  <head>
    <script>
      Class F1{
        constructor(name)
        {
          this.value = value;
        }
```

203

```
double(){
    this.value = this.value*2;
  }
}
const f2 = new F1(12);
f2.double();
console.log(f2.value);
</script>
</head>
</html>
```

-x-



Not a variable

It is a keyword

## this keyword

This refers to

Object method → object

refers to owner of function

- Alone → global object
- Function → global object
- Function (Strict mode) → undefined
- Event → refers to the element that received the event.
- Methods (call() apply() bind()) → Any object.

points to particular object.

cannot change value of this.