OOPS-2 : Classes & Constructors

AGENDA

- Arrow Functions
- Constructor Functions
- Classes
- Class based inheritance using Super, extends
- Static Methods of Class
- Private properties

* Constructor Functions :

- (Reusability)
  - Constructor function helps us to create multiple copies/objects with same structure, methods, properties.

- They act like a blueprint for creating instances of a particular type.

Constructor function ⟺ Pizza recipe

(Defines essential components of object (like dough, oven time, sauces)

+ allows for customization.

Pizza constructor function / (create pizza objects)

```
function Pizza (toppings, size, crustType) {

    this.toppings = toppings;
    this.size = size;
    this.crustType = crustType;

    this.describe = function () {
        console.log(
            this.toppings, size,)
    }
}
```

properties

method

Using same constructor functions.

```
Const order1 = new Pizza (['cheese',
                           'small',
                           'thin'])
```

```
Const order2 = new Pizza (['veg',
                           'large',
                           'thin'])
```

* Inside the function.

'this' refers to the new object that will be created when function is called with 'new' keyword.

'new' keyword will make 'this' keyword point to the newly created object.

Declare a

(naming a class)

```
Class MyClass {

    Constructor() {
    }

    method1() {
    }

}
```

- Creating & initializing properties created with the class
- Only one allowed

(function)

class method added to the class.

* Constructor → Automatically called when a new instance of a class is created

* properties – Variables that belong to an object.

* method – Methods are functions defined in a class.

Static methods & properties

- These are associated with class itself & not the instance of class.

- We can call a 'static' method, without creating instance of a class.

- eg. utility / common functions.

- needed by all instances.
- No dependency on any particular instance.

- Not tied to any instance
- Usage.
  - ↳ When you need (implement utility function that is present in the class).

Classes in ES6	Constructor function
- straightforward syntax for creating objects & using inheritance.	- Traditional way - Earlier versions.

(Prototypal inheritance)

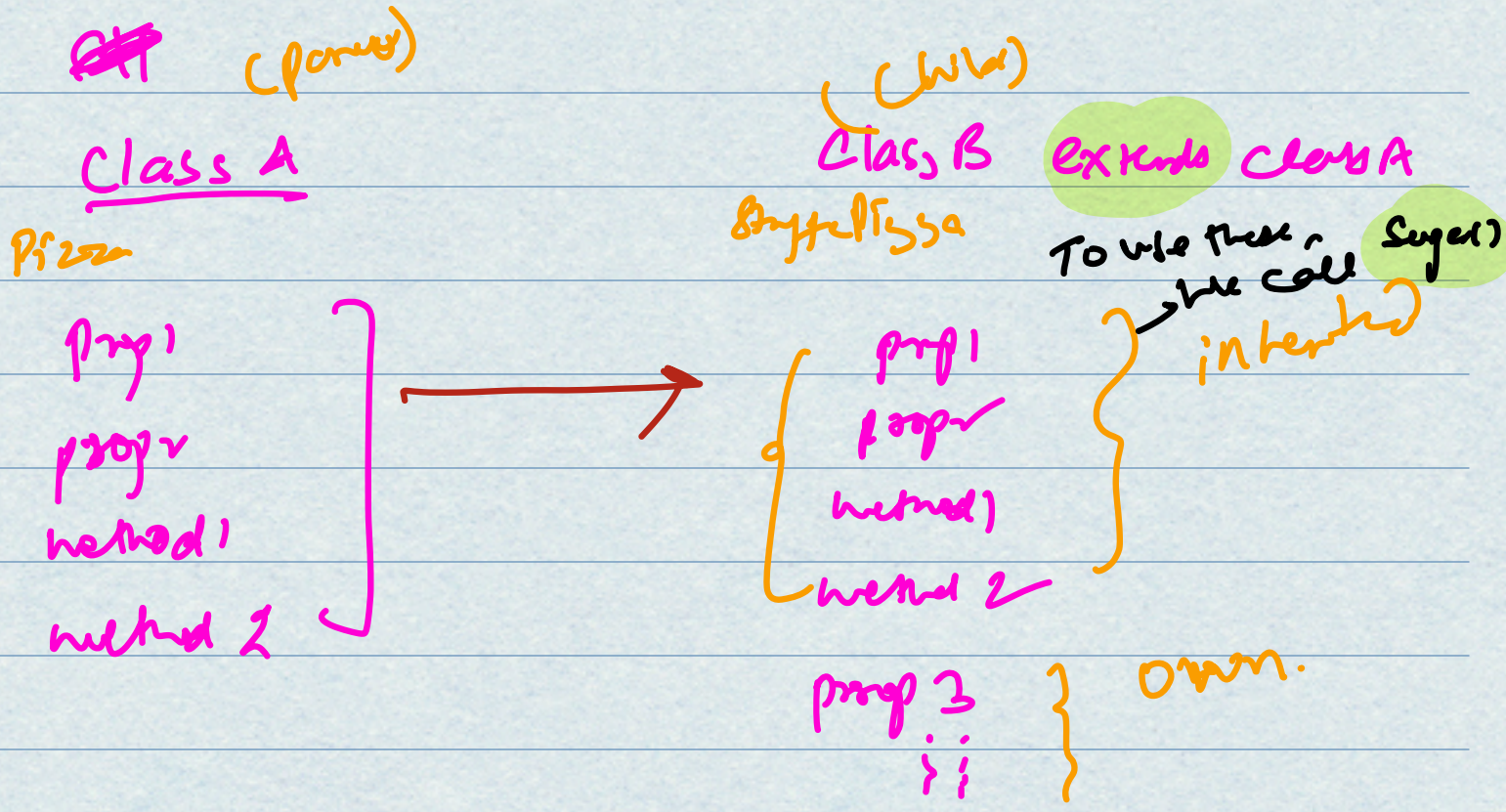
- Syntactical sugar over prototypal inheritance

Inheritance in JS

(super, extends) (class based syntax [ES-6].

↳ Allows a class to inherit properties

and methods from a different class



extends :
- to extend a class
- to create a class as a child of another class.

Super :

↳ Super keyword is used to access & call functions of the parent's object

super.describe(); // Methods

super(prop1, prop2); // Call parent's constructor.

(Classical Inheritance in JS.)

Paradigm of 'PROTOTYPAL INHERITANCE'