Class Declaration

- Configures a class that allows export.
- You can export the class from current module and use in any another module.
- Eager Loading [Loads automatically when module is loaded into memory].

Syntax:

```
class ClassName
{
}
```

Class Expression

- Configures local class
- It will not allow exporting
- It is accessible and used in the local module.
- Lazy Loading [Loads into memory only when requested]

Syntax:

```
var refName = class {
}
```

FAQ: What is the purpose of "export" keyword?

- Every member defined in a module is by-default local.
- To mark any member as global, so that it can be accessed outside module you have to use "export".
  Syntax:
  ```
  export class ClassName
  {
  }
  ```

FAQ: What is the purpose of "default" marker?

- Default is not a keyword.
- “default” is a marker, which is used to define that it is the default member to export.
- If a module contains multiple classes and you want only one class by default to export then
  mark it as “default”

Ex:

```
export default class Category
{
}
export class Product
{
}
```

Class Members:

- A class can contain only following members
  - Property
  - Method
  - Constructor
  - Accessor

FAQ: Can we define a variable in Class?

- No
- Class allows data only in property.

FAQ: What is difference between Property and Variable?

- Variable
  - It is immutable

    - Its structure can’t change dynamically according to state and situation.
    - Can’t have authorized access.
    - Class should allow dynamic; hence variables are not allowed in class.
- Property
    - It is mutable
    - Can change according to state and situation.
    - Can have authorized access.

FAQ: Can we define a function in Class?

- No
- Class can define functionality by using method.

FAQ: What is difference between function and method?

- Function
    - Can’t change dynamically
    - Always intended to return a value
    - It is used to build expression
    - Class can’t define expression, but it can use expression.
- Method
    - Method can change dynamically
    - It is not intended to return a value.
    - It configures a functionality, which can use expression.

Static and Non-Static Members

Static Members:

- Static uses continuous memory
- The memory allocated for first request will continue for other requests.

- Uses more memory
- Not safe [Memory Leaks]
- Static members in a class are accessible with-in or outside the class by using “ClassName”.

Dynamic [Non-Static]

- Uses discreet memory
- Disconnected in access
- Memory is newly allocated for every object.
- Destroys and re-allocated on request.
- Non-Static members are accessible with-in the class by using “this” keyword and outside the class by using an instance of class.

Ex:

```
class Demo
{
 static s = 0;
 n = 0;
 constructor(){
  Demo.s = Demo.s + 1;
  this.n = this.n + 1;
 }
 Print(){
  console.log(`s=${Demo.s} n=${this.n}`);
 }
```

```
}
let obj1 = new Demo;
obj1.Print();
let obj2 = new Demo;
obj2.Print();
let obj3 = new Demo;
obj3.Print();
```

Access Modifiers

- To handle code level security
- Restrict the accessibility of member.
    - public
    - private
    - protected

Public

- It is accessible with-in the class.
- It is accessible outside the class by using base class object and derived class object.

Private:

- It is accessible with-in the class
- It is not accessible outside the class.

Protected:

- It is accessible with-in the class

- It is accessible outside the class only by using derived class object with-in derived class.

Properties

Methods

Constructors

Accessors