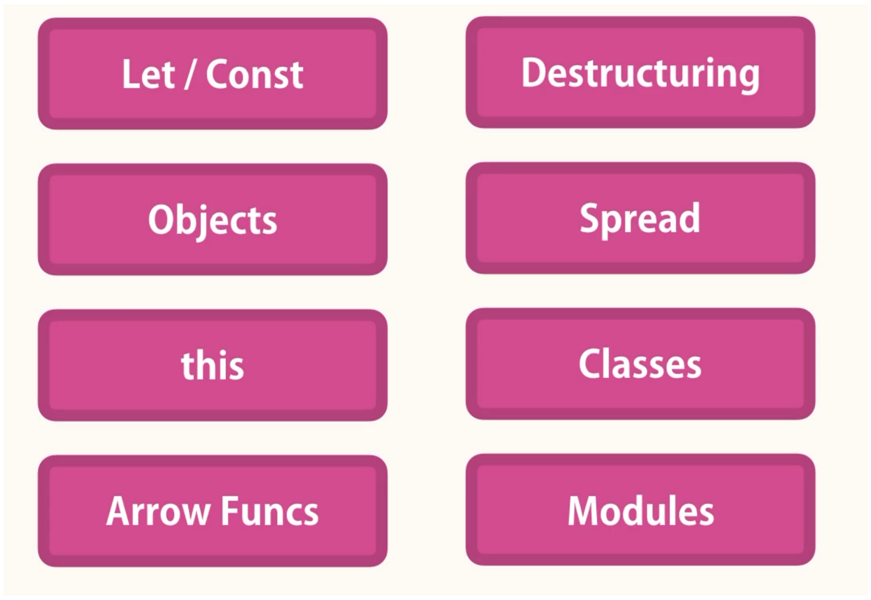
ES6



1. **<u>Let vs Var vs Const</u>**

```
// var -> function
// let -> block
// const -> block

function sayHello(){
    for(var i=0; i< 5; i++)
    {
        console.log(i);
    }

  // i value is access from the loop
    console.log(i);
}
```

```
// var -> function
// let -> block
// const -> block
```

```
function sayHello(){
    for(var i=0; i< 5; i++)
    {
        console.log(i);
    }

  // i value is access from the loop
    console.log(i);
}
```

```
function sayHello(){
    for(let i=0; i< 5; i++)
    {
        console.log(i);
    }
    console.log(i);
}

// error
```

```
let x = 2;
x= 3;

Uncaught TypeError: Assignment to constant variable.
```

2. **Object**

```
const person = {
    name : 'Arkar',
    walk:function(){},
    talk(){}
}

const person = {
    name : 'Arkar',
    walk(){},
    talk(){}
}

// access
person.talk();
person['name'] = 'Htet Myint';
person.name = 'Htet Myint';
```

3. **This keyword**

```
// Return the current object
const person = {
    name : 'Arkar',
    walk(){
        console.log(this);
    }
}

person.walk();

// Show the person object in the console
//The java script alway that way

const walk = person.walk;
console.log(walk);

const walk = person.walk;
walk();
// undefined at console
//*Note
//Object of the object this keywork is call to Global object
// Return Window object
```

4. **Binding this**

```
// Java Script function are object

const walk = person.walk.bind(person);
walk();
```

5. **Arrow Functions**

```
// 1
const square = function(number){
   return number*number;
}

// 1.1 single parameter
const square = number => {
    return number*number;
 }

// 1.2
```

```
const square = number =>  number*number;
console.log(square(5));


// Another exaple
const jobs = [
    {id : 1 , isActive:true},
    {id : 2 , isActive:true},
    {id : 3 , isActive:false},
];

const activeJobs = jobs.filter(function(job){return job.isActive});
const activeJobs = jobs.filter(job => job.isActive);
```

6. Arrow functions with this

```
const person = {
    talk() {
        console.log('this',this)
    }
}
// output the current object
person.talk();


// out put the window object
// setTimeout function is not the part of person object
const person = {
    talk() {
        setTimeout(function(){
            console.log('this',this)
        },1000)

    }
}

person.talk();

// var self = this
// console = self
```

```
// change to arrow function
const person = {
    talk() {
        setTimeout( () => {
            console.log('this',this)
        },1000)

    }
}
person.talk();
```

7. **Array.map Method**

```
const colors = ['red','green','blue'];
colors.map( function(color){
    console.log(color);
});

// change arrow function
colors.map( color => console.log(color));

//template literals
const items = colors.map( color => `<li>${color}</li>`);
console.log(items);
```

8. **Object Destructuring**

```
const address = {
    street : '',
    city : '',
    country : '',
}

// multiple place 'address.'
const street = address.street;
const city = address.city;
const county = address.country;

//equivalent
const {street, city, country} = address;

// different name
const {street : st, city, country} = address;
```

9. **Spread operator**

```
const first = [1,2,3];
const second = [4,5,6];

const combined = first.concat(second);
const combined = [...first]
const combined = [...first,...second]
const combined = [...first,'a',...second,'b']
```

```
const first = {name:'Ar Kar'};
const second = {job:'Instructor'};

const combined = {...first,...second,locaiont:'YGN'};
console.log(combined);

const clone = {...first};
```