

Coding for Teens

JS

Exploring

#7 JavaScript ES6

ECMAScript 6

- Latest standardized version of ECMAScript 5.
- Includes many new features.
- Not fully supported in browser yet.

Let Keyword

```
var x = 5  
var x = 10  
console.log(x)
```

=====

```
let y = 2  
let y = 4  
console.log(y)
```

Constant

```
const pi = 3.14;  
pi = 10  
console.log(pi);
```

// Error

Constant

```
const pi = 3.14;
```

```
function luasLingkaran(r){  
    const pi = 10;  
    console.log('Luas='+pi*r*r);  
}
```

```
console.log(pi);           // 3.14  
luasLingkaran(5);         // Luas=250
```

Template String

```
let halo = `Halo
```

```
  Dunia!`
```

```
console.log(halo);
```

```
// Use tick (`) below Esc!
```

Template String

```
function halo(nama) {  
  console.log('Halo, aku '+nama);}
```

```
function hai(nama) {  
  console.log(`Hai, aku ${nama}`);  
  console.log(`Usiaku ${7*3}`);}
```

```
halo('Andi');  
// Halo, aku Andi
```

```
hai('Arif');  
// Hai, aku Arif  
// Usiaku 21
```

String Methods

repeat & includes

```
let x = 'halo';
```

```
console.log(x.repeat(5));  
// halohalohalohalohalo
```

```
console.log(x.includes('lo'));  
// true
```


String Methods

startsWith & endsWith

```
let x = 'halo';
```

```
console.log(x.startsWith('ha'));  
console.log(x.startsWith('lo'));  
console.log(x.startsWith('lo', 2));
```

```
console.log(x.endsWith('lo'));  
console.log(x.endsWith('ha'));  
console.log(x.endsWith('ha', x.length-2));
```

Spread Operator

```
let buah =  
['apel', 'duku', 'pir'];
```

```
console.log(buah);  
// ['apel', 'duku', 'pir']
```

```
console.log(...buah);  
// apel duku pir
```

Spread Operator

```
let no1 = [1, 2, 3];  
let no2 = [no1, 4, 5, 6];  
let no3 = [...no1, 7, 8, 9];
```

```
console.log(no2);  
// [[1, 2, 3], 4, 5, 6]
```

```
console.log(no3);  
// [1, 2, 3, 7, 8, 9]
```

Spread Operator

```
let angka = [1,2,3];  
function jumlah(x,y,z){  
    console.log(x+y+z);  
}
```

```
jumlah(angka);  
// 1,2,3undefinedundefined
```

```
jumlah(angka[0],angka[1],angka[2]);  
// 6
```

```
jumlah(...angka);  
// 6
```

Default Parameter

```
function kuadrat(x=5){  
    console.log(x*x);  
}
```

```
kuadrat();           // 25  
kuadrat(10);         // 100
```

Arrow Function

```
let halo = function(){  
  console.log('Halo Dunia!')  
}
```

```
let hai = () => {  
  console.log('Hai Hacker!')  
}
```

```
let alo = () => console.log('Aloha!')
```

```
halo(); hai(); alo();
```

Arrow Function

Return function

```
let pi = () => {  
  return 3.14  
}
```

```
let g = () => 9.8
```

```
console.log(pi());  
console.log(g());
```

Arrow Function

With a parameter

```
let halo = (nama) => {  
  console.log(`Halo ${nama}`)  
}
```

```
halo('Andi');
```

```
let hai =  
nama => console.log(`Hai ${nama}`)
```

```
hai('Budi');
```


Arrow Function

With 2 parameters

```
let halo = (x,y) => {  
  console.log(`Halo ${x} ${y}`)  
}
```

```
halo('Andi',21);
```

Callback Function

#1

```
let x = function () {  
    console.log('Hai ini X!');  
};
```

```
let y = function (callback) {  
    console.log('Halo ini Y!');  
    callback();  
};
```

```
y(x);
```

Callback Function

#2

```
let x = () => {  
  console.log('Hai ini X!');  
};
```

```
let y = (callback) => {  
  console.log('Halo ini Y!');  
  callback();  
};
```

```
y(x);
```

Callback Function

#3 application without callback function

```
let hitung = (no1, no2, op) => {  
  if(op=='kali'){  
    return no1*no2  
  };  
  if(op=='bagi'){  
    return no1/no2  
  };  
};  
console.log(hitung(2, 3, 'kali'));
```

Callback Function

#3 application with callback function

```
let kali = (x, y) => {  
  return x * y;  
}  
let bagi = (x, y) => {  
  return x / y;  
}  
let hitung = (no1, no2, op) => {  
  return op(no1, no2);  
};  
  
console.log(hitung(2, 3, kali));
```

Callback Function

#3 application with callback function

```
let kali = (x, y) => x * y;
```

```
let bagi = (x, y) => x / y;
```

```
let hitung =  
(no1, no2, op) => op(no1, no2);
```

```
console.log(hitung(2, 3, kali));
```

Callback Function

#4 insert anonymous function

```
let hitung = (no1, no2, op) => {  
    return op(no1, no2);  
};
```

```
console.log(hitung(2, 3, function(x,y) {  
    return x + y;  
}));
```

```
/*  
console.log(hitung(2, 3, (x,y) => x + y))  
*/
```

Array Filtering

```
var w = [0,1,2,3,4,5]
```

```
var x = w.filter((val) => val !== 2);
```

```
var y = w.filter((val) => val % 2 === 0);
```

```
var z = w.filter((val) => val % 2 !== 0);
```

```
console.log(x);
```

```
console.log(y);
```

```
console.log(z);
```

```
// ((val) => val % 2 !== 0) artinya
```

```
// ((val) => {return val % 2 !== 0})
```


Array Mapping

```
var w = [1,4,9,16,25]
```

```
var x = w.map(Math.sqrt);
```

```
var y = w.map((val)=> val * 2);
```

```
var z = w.map((val)=> val !== 9);
```

```
console.log(x);
```

```
console.log(y);
```

```
console.log(z);
```

Array Mapping

```
var orang = [  
  {nama : "Andi", marga: "Hasibuan"},  
  {nama : "Budi", marga: "Sinaga"},  
  {nama : "Caca", marga: "Pasaribu"}  
];  
function namaLengkap(item, index) {  
  var fullname = [item.nama,item.marga].join(" ");  
  return fullname;  
}  
function tesMap() {  
  console.log(orang.map(namaLengkap));  
  console.log(orang.map(namaLengkap)[0]);  
  console.log(orang.map(namaLengkap)[1]);  
  console.log(orang.map(namaLengkap)[2]);  
}  
tesMap()
```

Object Literal *properties*

```
let merk = 'Yamaha';  
let tahun = 2015;
```

```
let mio = {  
  merk: merk,  
  prod: tahun  
}
```

```
let vixion = {  
  merk, tahun  
}
```

```
console.log(mio);  
console.log(vixion);
```

Object Literal *method*

```
let mio = {  
  kualitas: function(x){  
    return `Mutu ${x}`;  
  }  
}
```

```
let vixion = {  
  kualitas(x){  
    return `Performa ${x}`;  
  }  
}
```

```
console.log(mio.kualitas('Oke banget!'));  
console.log(vixion.kualitas('Juara!'));
```

Set

```
let nama = new Set();
```

```
nama.add('Adi').add('Budi').add('Adi');  
console.log(nama);  
console.log(nama.size);
```

```
nama.delete('Adi');  
console.log(nama);  
console.log(nama.size);
```

```
nama.clear();  
console.log(nama);  
console.log(nama.size);
```

*Set is kind of
Array that has no
duplicate items.*

*To access Set
Element, simply
convert it to
Array first.*

Array to Set

```
let id = ['Ali', 'Bona', 'Ali'];  
let nama = new Set(id);  
  
console.log(nama);  
console.log(nama.size);
```

Array to Set Then Set to Array

```
let angka = [1,2,3,4,1,2,5,6];  
console.log(angka);
```

```
let nomor = new Set(angka);  
console.log(nomor);
```

```
let arrayAngka = [...nomor]  
console.log(arrayAngka);
```

Destructuring *Array*

```
let buah = [  
  'Apel',  
  'Duku',  
  'Leci'  
]
```

```
var [ x, y, z ] = buah;
```

```
console.log(x);  
console.log(y);  
console.log(z);
```


Destructuring *Object*

```
let andi = {  
  nama: 'Andi',  
  usia: 24,  
  job: 'PNS'  
}
```

```
var { nama, usia, job } = andi;
```

```
console.log(nama);  
console.log(usia);  
console.log(job);
```

Coding for Teens

JS

Exploring

#7 JavaScript ES6