
Eavascript

ES6

ES6

Javascript introduit par Netscape

Puis ECMA International pour la standardisation

ES6 == EcmaScript 6 (2015) ES7 (2016) ES8(2017)

⇒ > le code javascript devient plus simple

Mais pas supporter par la plupart des navigateur

=>Babel

ES6

Javascript

```
var a=function(x,y){  
    return x+y;  
}
```

ES6

```
const a=(x,y)=>x+y;
```

ES6: destructring

Javascript

```
1  const etudiant={
2      nom: "Baddi",
3      prenom: "Ahmed",
4      age: 24
5  }
6  const nom=etudiant.nom;
7  const prenom=etudiant.prenom;
8  const age=etudiant.age;
```

ES6

```
1  const etudiant={
2      nom: "Baddi",
3      prenom: "Ahmed",
4      age: 24
5  }
6
7  const {nom, prenom, age}=etudiant;
```

ES6: proprietes des objets

```
1  const a="nom";  
2  const b="pre";  
3  const etudiant={  
4      [a]: "Baddi",  
5      [b+a]: "Ahmed",  
6      age: 24  
7  }  
8  console.log(etudiant.prenom);
```

ES6: proprietes des objets

```
1  const nom="Baddi";
2  const prenom="Ahmed";
3  const age=24;
4  const etudiant={
5      nom: nom,
6      prenom: prenom,
7      age: age
8  }
9  //ES6
10 const etudiant={
11     nom,
12     prenom,
13     age
14 }
15
```

ES6: template Strings

```
1  const nom="Baddi";
2  const prenom="Ahmed";
3  const age=24;
4  let s= "Mon nom est"+nom+"j'ai "+age+"
      ans";
5  //ES6
6  let s=`Mon nom est: ${nom} j'ai ${age}
      ans`;
```

ES6: Arguments par défaut

```
1  function cal(a=0,b=1){  
2      return a/b;  
3  }  
4  
5  calc(); // 0  
6  calc(4); // 4  
7  calc(4,8); // 0.5
```


ES6: Arrow function

```
1  function somme(a,b){  
2      return a+b;  
3  }  
4  //ES6  
5  const somme=(a,b)=> {  
6      return a+b  
7  }  
8  
9  const somme=(a,b)=> a+b;
```

ES6: Closures

```
1 function f1(){
2     var a="Bonjour";
3     function f2(){
4         console.log(a);
5     }
6     return f2;
7 }
8 var f=f1();
9 f();
```

ES6: Closures

```
1 function f1(){
2     var a="Bonjour";
3     function f2(){
4         console.log(a);
5     }
6     return f2;
7 }
8 var f=f1();
9 f();
```

```
//ES6
const f1={()=>{
    const a="Bonjour";
    const f2={()=>{
        console.log(a);
    }
}
var f=f1();
f();
```

ES6: Currying

Transformer une fonction a plusieurs variable en plusieurs fonctions a une seule variable.

```
1  const prod1=(a,b)=> a*b;  
2  prod1(3,4); // 12  
3  
4  const prod2=(a)=>(b)=>a*b;  
5  prod2(3); // ?  
6  prod2(3)(4); // ?
```

ES6: Currying

Transformer une fonction a plusieurs variable en plusieurs fonctions a une seule variable.

```
1  const prod1=(a,b)=> a*b;  
2  prod1(3,4); // 12  
3  
4  const prod2=(a)=>(b)=>a*b;  
5  prod2(3); // ?  
6  prod2(3)(4); // ?
```

ES6: Compose

```
1  const comp = (f, g) => (a) => f(g(a));  
2  let somme = (n) => n+1;  
3  comp(somme, somme)(3); // ?
```

Array

```
1  const a=[1,4,3,8];
2  let b=a.forEach((n)=>{
3      n*2;
4  });
5  console.log(b);// undefined
6  let b=a.map((n)=> n*2);
7  console.log(b);// [2,8,6,16]
8
9  let c=a.filter((n)=> n%2 === 0);
10 console.log(c);// [4,8]
11
12 let d=a.reduce((s,n)=> s+n,2);
13 console.log(d);// 18
14
```

Objets : references

```
1  let a={val: 3};  
2  let b=a;  
3  let c={val: 3};  
4  console.log(a===b); //true  
5  console.log(a===c); //false  
6  a.val=2;  
7  console.log(a.val); //2  
8  console.log(b.val); //2  
9  console.log(c.val); //3
```


Objets: contexte

```
1  const obj={
2      a:function(){
3          console.log(this);
4      }
5  }
6  obj.a();//obj
7  console.log(this);// window
8  function a(){
9      console.log(this);
10 }
11 a()// window
```

Objets: instantiation

```
class Homme{  
  constructor(nam, age){  
    this.nom=nom;  
    this.age=age;  
  }  
  hi(){  
    console.log(`Je suis ${this.nom}`);  
  }  
}  
let h=new Homme("yakoubi",22);
```

Objets: instantiation

```
class Etudiant extends Homme{
  constructor(nom, age, note){
    super(nom, age);
    this.note=note;
  }
  getNote(){
    console.log(`Note: ${this.note}`);
  }
}
let h=new Homme("yakoubi",22,17);
```