

TP : Découverte de JavaScript Moderne (ES6+)

Objectif général

Apprendre les bases pratiques du langage JavaScript moderne à travers des exercices guidés : *variables, fonctions, objets, tableaux, callbacks, promesses, async/await, etc.*

Installation et préparation de l'environnement

1. Installer Node.js

- Va sur <https://nodejs.org>
- Télécharge la version **LTS** et installe-la.
- Vérifie :
 - node -v
 - npm -v

2. Créer ton projet

```
mkdir tp-js-base
cd tp-js-base
npm init -y
```

3. Créer un fichier principal

Crée un fichier `index.js` :

```
touch index.js
```

Variables et portée

Exercice 1

1. Déclare trois variables :

```
var x = 5
let y = 10
const z = 15
```

2. Essaie de réassigner chacune d'elles.
3. Observe les erreurs et explique pourquoi.

```
/Users/vesion2/Documents/tp-js-base/index.js:6
```

```
z = 4
^
```

```
TypeError: Assignment to constant variable.|
```

Explication : On peut pas réassigner une constante.

Exercice 2

Teste la portée :

```
function testScope() {
  if (true) {
    var a = "var visible partout";
    let b = "let visible ici seulement";
  }
  console.log(a);
  console.log(b); // Pourquoi erreur ?
}
testScope();
```

Explication : var est une variable local dans une fonction alors que let est local entre {}

Fonctions classiques et fléchées

Exercice 3

Crée deux fonctions qui saluent un utilisateur :

```
function sayHello(name) {
    return `Bonjour ${name}`;
}

const sayHelloArrow = (name) => `Bonjour ${name}`;
```

Compare leur comportement et explique la différence.

Reponse : Les deux saluent pareil, mais sayHello est une fonction classique hoistée avec son propre this, alors que sayHelloArrow est une fonction fléchée non hoistée qui hérite du this externe.

Exercice 4

Teste la portée de this :

```
const person = {
    name: "Sara",
    sayHello: function () {
        console.log("Bonjour " + this.name);
    },
    sayHelloArrow: () => {
        console.log("Bonjour " + this.name);
    },
};

person.sayHello();
person.sayHelloArrow();
```

Test :

```
[Running] node "/Users/vesion2/Documents/tp-js-base/index.js"
Bonjour Sara
Bonjour undefined

[Done] exited with code=0 in 0.149 seconds
```

Explication : sayHello utilise une fonction classique ,this fait référence à l'objet person.

sayHelloArrow est une fonction fléchée , elle n'a pas de this propre et hérite du contexte global, donc this.name est undefined

Import / Export de modules (ES6)

Apprendre à séparer le code en plusieurs fichiers pour mieux l'organiser et le réutiliser.

Étape 1 : Créer un module

Crée un fichier `mathUtils.js` :

```
// mathUtils.js
export const PI = 3.14;

export function carre(x) {
    return x * x;
}

// Export par défaut
```

```
export default function message() {
    console.log("Module mathUtils chargé !");
}
```

Étape 2 : L'utiliser dans ton script principal

Dans ton `index.js` :

```
// index.js
import message, { PI, carre } from "./mathUtils.js";

message(); // Module mathUtils chargé !
console.log("PI =", PI);
console.log("Carré de 5 =", carre(5));
```

Astuce :

Ajoute dans ton `package.json` :

```
{ "type": "module" }
```

Pour que Node.js reconnaisse la syntaxe `import/export`.

```
[Running] node "/Users/vesion2/Documents/tp-js-base/index.js"
Module mathUtils chargé !
PI = 3.14
Carré de 5 = 25

[Done] exited with code=0 in 0.16 seconds
```

Tableaux et méthodes modernes

Exercice 5 – Manipulation de base

```
const fruits = ["pomme", "banane", "orange"];
fruits.push("kiwi");
fruits.pop();
console.log(fruits);
```

```
[Running] node "/Users/vesion2/Documents/tp-js-base/index.js"
[ 'pomme', 'banane', 'orange' ]

[Done] exited with code=0 in 0.151 seconds
```

Exercice 6 – map, filter, reduce

```
const nums = [1, 2, 3, 4, 5];

// Multiplie chaque nombre par 2
console.log(nums.map(n => n * 2));

// Garde seulement les nombres pairs
console.log(nums.filter(n => n % 2 === 0));

// Calcule la somme totale
console.log(nums.reduce((sum, n) => sum + n, 0));
```

```
[Running] node "/Users/vesion2/Documents/tp-js-base/index.js"
[ 2, 4, 6, 8, 10 ]
[ 2, 4 ]
15

[Done] exited with code=0 in 0.166 seconds
```

Exercice 7 – find, some, every

```
console.log(nums.find(n => n > 3));
console.log(nums.some(n => n < 0));
console.log(nums.every(n => n > 0));
```

```
[Running] node "/Users/vesion2/Documents/tp-js-base/index.js"
[ 2, 4, 6, 8, 10 ]
[ 2, 4 ]
15
4
false
true

[Done] exited with code=0 in 0.151 seconds
```

Objets et déstructuration

Exercice 8

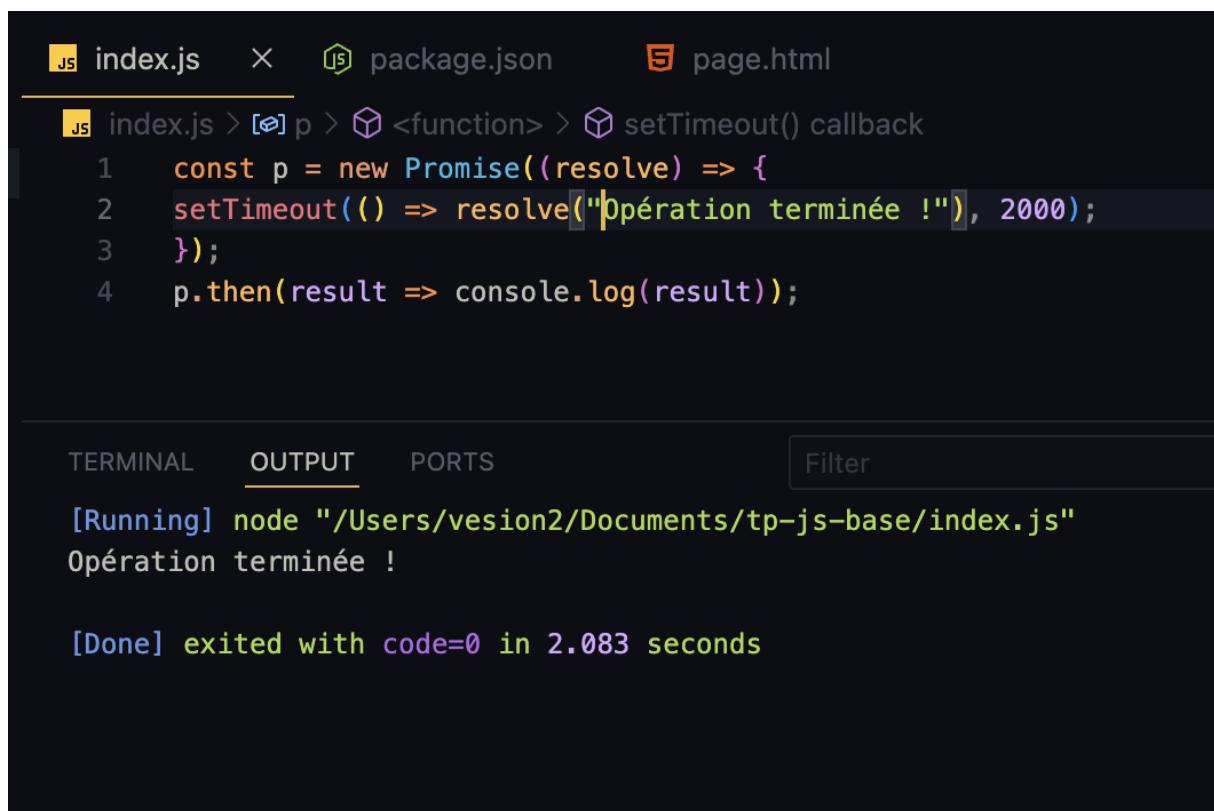
```
const user = { id: 1, name: "Ali", city: "Rabat" };

// Déstructuration
const { name, city } = user;
console.log(`${name} habite à ${city}`);

// Renommage
const { name: fullName, ...rest } = user;
console.log(fullName);
console.log(rest);
```

```
[Running] node "/Users/vesion2/Documents/tp-js-base/index.js"
Ali habite à Rabat
Ali
{ id: 1, city: 'Rabat' }
```

```
[Done] exited with code=0 in 0.152 seconds
```



index.js package.json page.html

index.js > [promise] p > <function> > setTimeout() callback

```
1 const p = new Promise((resolve) => {
2   setTimeout(() => resolve("Opération terminée !"), 2000);
3 }
4 p.then(result => console.log(result));
```

TERMINAL OUTPUT PORTS Filter

[Running] node "/Users/vesion2/Documents/tp-js-base/index.js"
Opération terminée !

[Done] exited with code=0 in 2.083 seconds

Exercice 10 – async/await

TP. ES6 / JS Moderne

```

index.js  X  package.json  page.html

index.js > getUsers > res
1  async function getUsers() {
2    try {
3      const res = await
4      fetch("https://jsonplaceholder.typicode.com/users");
5      const data = await res.json();
6      console.log(data);
7    } catch (e) {
8      console.error("Erreur :", e.message);
9    }
10   }
11   getUsers();

[{"id": 1, "name": "Romain Grosjean", "username": "Romain.Grosjean", "email": "Romain.Grosjean@renault.com", "address": {"street": "Boulevard de la Marne", "suite": "100", "city": "Paris", "zipcode": "75009", "geo": {"lat": 48.8584, "lon": 2.3514}}, "phone": "+33 1 40 20 80 00", "website": "http://www.renault.com"}, {"id": 2, "name": "Clementina DuBuque", "username": "Moriah.Stanton", "email": "Rey.Padberg@karina.biz", "address": {"street": "Kattie Turnpike", "suite": "Suite 198", "city": "Lebsackbury", "zipcode": "31428-2261", "geo": {"lat": 37.2921, "lon": -121.5246}}, "phone": "024-648-3804", "website": "ambrose.net", "company": {"name": "Hoeger LLC", "catchPhrase": "Centralized empowering task-force", "bs": "target end-to-end models"}}, {"id": 3, "name": "Patricia Lebsack", "username": "Markus.Hill", "email": "Patricia.Lebsack@agusti.net", "address": {"street": "Tanner Haven", "suite": "1545", "city": "South Jennifer", "zipcode": "92704-6520", "geo": {"lat": 39.9625, "lon": -77.3438}}, "phone": "024-417-6543", "website": "http://agusti.net"}, {"id": 4, "name": "Kathryn Muller", "username": "Doris.Schmidt", "email": "Kathryn.Muller@reynolds.ca", "address": {"street": "Tanner Haven", "suite": "1545", "city": "South Jennifer", "zipcode": "92704-6520", "geo": {"lat": 39.9625, "lon": -77.3438}}, "phone": "024-417-6543", "website": "http://reynolds.ca"}, {"id": 5, "name": "Nicholas Runte", "username": "Audrey.Tremblay", "email": "Nicholas.Runte@fritsch.com", "address": {"street": "Tanner Haven", "suite": "1545", "city": "South Jennifer", "zipcode": "92704-6520", "geo": {"lat": 39.9625, "lon": -77.3438}}, "phone": "024-417-6543", "website": "http://fritsch.com"}, {"id": 6, "name": "Nicholas Runte", "username": "Audrey.Tremblay", "email": "Nicholas.Runte@fritsch.com", "address": {"street": "Tanner Haven", "suite": "1545", "city": "South Jennifer", "zipcode": "92704-6520", "geo": {"lat": 39.9625, "lon": -77.3438}}, "phone": "024-417-6543", "website": "http://fritsch.com"}, {"id": 7, "name": "Nicholas Runte", "username": "Audrey.Tremblay", "email": "Nicholas.Runte@fritsch.com", "address": {"street": "Tanner Haven", "suite": "1545", "city": "South Jennifer", "zipcode": "92704-6520", "geo": {"lat": 39.9625, "lon": -77.3438}}, "phone": "024-417-6543", "website": "http://fritsch.com"}, {"id": 8, "name": "Nicholas Runte", "username": "Audrey.Tremblay", "email": "Nicholas.Runte@fritsch.com", "address": {"street": "Tanner Haven", "suite": "1545", "city": "South Jennifer", "zipcode": "92704-6520", "geo": {"lat": 39.9625, "lon": -77.3438}}, "phone": "024-417-6543", "website": "http://fritsch.com"}, {"id": 9, "name": "Nicholas Runte", "username": "Audrey.Tremblay", "email": "Nicholas.Runte@fritsch.com", "address": {"street": "Tanner Haven", "suite": "1545", "city": "South Jennifer", "zipcode": "92704-6520", "geo": {"lat": 39.9625, "lon": -77.3438}}, "phone": "024-417-6543", "website": "http://fritsch.com"}, {"id": 10, "name": "Nicholas Runte", "username": "Audrey.Tremblay", "email": "Nicholas.Runte@fritsch.com", "address": {"street": "Tanner Haven", "suite": "1545", "city": "South Jennifer", "zipcode": "92704-6520", "geo": {"lat": 39.9625, "lon": -77.3438}}, "phone": "024-417-6543", "website": "http://fritsch.com"}]
[Done] exited with code=0 in 0.277 seconds

```

Fonctions avancées et opérateurs modernes

Exercice 11 – Template literals

```

26
27
28 
29 const name = "Nadia";
30 const hour = new Date().getHours();
31 console.log(`Bonjour ${name}, il est ${hour}h`);
```

TERMINAL **OUTPUT** PORTS

Filter

```
SyntaxError: Unexpected token '&'  
Node.js v24.10.0
```

```
[Done] exited with code=1 in 0.063 seconds
```

```
[Running] node "/Users/vesion2/Documents/tp-js-base/index.js"  
Bonjour Nadia, il est 11h
```

```
[Done] exited with code=0 in 0.052 seconds
```

Exercice 12 – Spread / Rest

```

28 const arr1 = [1, 2];
29 const arr2 = [...arr1, 3, 4];
30 console.log(arr2);
31
32
33 4 / 2
34
35 function sum(...numbers) {
36   return numbers.reduce((a, b) => a + b, 0);
37 }
38 console.log(sum(1, 2, 3, 4));
```

TERMINAL **OUTPUT** PORTS

Filter

Code

```
[Running] node "/Users/vesion2/Documents/tp-js-base/index.js"  
[ 1, 2, 3, 4 ]  
10
```

```
[Done] exited with code=0 in 0.064 seconds
```

Exercice 13 – Optional chaining et Nullish coalescing

```
28 const settings = { theme: null };
29 console.log(settings.theme ?? "light"); // light
30 const user2 = { profile: { email: "x@y.com" } };
31 console.log(user2.profile?.email); // x@y.com
32 console.log(user2.address?.city); // undefined
```

TERMINAL

OUTPUT

PORTS

Filter

Code

```
[Done] exited with code=1 in 0.062 seconds
```

```
[Running] node "/Users/vesion2/Documents/tp-js-base/index.js"
light
x@y.com
undefined
```

```
[Done] exited with code=0 in 0.056 seconds
```

Gestion des produits

Objectif

Créer un script qui :

1. Définit une liste de produits (nom, prix, date d'expiration).
2. Filtre les produits non expirés.
3. Calcule la somme totale.
4. Affiche le résultat formaté.

TP. ES6 / JS Moderne

```
23
24  const produits = [
25    { nom: "Thon", prix: 10, expireLe: "2025-12-01" },
26    { nom: "Lait", prix: 5, expireLe: "2024-01-01" },
27    { nom: "Jus", prix: 8, expireLe: "2026-02-15" },
28  ];
29  const aujourdHui = new Date();
30  const valides = produits.filter(p => new Date(p.expireLe) >
31  aujourdHui);
32  const total = valides.reduce((s, p) => s + p.prix, 0);
33  console.log("Produits valides :", valides);
34  console.log("Total :", total, "DH");
```

TERMINAL

OUTPUT

PORTS

Filter

Code

[Running] node "/Users/vesion2/Documents/tp-js-base/index.js"

Produits valides : [

```
  { nom: 'Thon', prix: 10, expireLe: '2025-12-01' },
  { nom: 'Jus', prix: 8, expireLe: '2026-02-15' }
]
```

Total : 18 DH

[Done] exited with code=0 in 0.057 seconds