01_Javascript et Node.js

Introduction à JavaScript et Node.js

Javascript

JavaScript est un langage de programmation interprété et orienté objet, principalement utilisé pour le développement web côté client. Il permet de rendre les pages web interactives et dynamiques. Voici quelques points clés :

1. **Langage de Script** : JavaScript est souvent utilisé pour écrire des scripts qui s'exécutent dans le navigateur web. Ces scripts peuvent manipuler le DOM (Document Object Model) pour modifier le contenu et l'apparence des pages web en temps réel.
2. **Côté Client** : Historiquement, JavaScript était exclusivement exécuté côté client, c'est-à-dire dans le navigateur de l'utilisateur. Cela permettait d'améliorer l'expérience utilisateur sans nécessiter des allers-retours constants avec le serveur.
3. **Bibliothèques et Frameworks** : JavaScript dispose d'une riche écosystème de bibliothèques (comme jQuery, Lodash) et de frameworks (comme React, Angular, Vue.js) pour faciliter et accélérer le développement web.

Exemple de code :
Modifier le contenu d'une page web

```
<!DOCTYPE html>
<html>
<head>
    <title>Exemple JavaScript</title>
</head>
<body>
    <h1 id="titre">Bonjour, monde!</h1>
    <button onclick="changerTexte()">Changer le texte</button
>
```

```
    <script>
        function changerTexte() {
            document.getElementById('titre').innerText = 'Bon
jour, JavaScript!';
        }
    </script>
</body>
</html>
```

Node.js

Node.js est un environnement d'exécution pour JavaScript qui permet d'exécuter du code JavaScript côté serveur. Voici quelques points importants :

1. **Côté Serveur** : Contrairement à JavaScript exécuté dans le navigateur, Node.js permet d'exécuter du code JavaScript sur le serveur, ce qui ouvre des possibilités pour le développement d'applications web complètes utilisant un seul langage de programmation.
2. **V8 Engine** : Node.js est basé sur le moteur JavaScript V8 de Google (utilisé dans Chrome), ce qui le rend très performant.
3. **Asynchrone et Événementiel** : Node.js est conçu pour être non-bloquant et basé sur les événements. Cela signifie qu'il peut gérer un grand nombre de connexions simultanées avec un haut degré d'efficacité, ce qui le rend idéal pour les applications en temps réel (comme les chats, les jeux en ligne, etc.).
4. **NPM (Node Package Manager)** : Node.js dispose de NPM, un gestionnaire de paquets qui contient des milliers de modules que vous pouvez utiliser pour ajouter des fonctionnalités à vos applications. Quelques exemples de modules populaires incluent Express (un framework web), Mongoose (pour travailler avec MongoDB), et Socket.io (pour les applications en temps réel).

Exemple de code :
Création d'un serveur HTTP en Node.js

```
const http = require('http');

const server = http.createServer((req, res) => {
    res.statusCode = 200;
    res.setHeader('Content-Type', 'text/plain');
    res.end('Hello, Node.js!\n');
});

const port = 3000;
server.listen(port, () => {
    console.log(`Le serveur tourne sur http://localhost:${por
t}/`);
});
```