בס"ד

קורס NodeJS תשפה

מבוא

NodeJS היא סביבת ריצה לקוד javascript, המיועדת להרצת קוד בserver.

NodeJS רצה על מנוע V8 של google chrome מחוץ לדפדפן. הוא open source וcross platform ולכן פופולרי מאד בשנים האחרונות בפיתוח מערכות backend מורכבות.

אפליקציה של node רצה על single process, בלי ליצור thread חדש לכל בקשה. כדי למנוע מהthread להיתקע על ביצוע פעולות כמו I/O, פעולות אלו מבוצעות בצורה אסינכרונית – הפעולה נשלחת לביצוע ע"י הnetwork, מערכת הקבצים, הDB או כל ספק חיצוני אחר. כשביצוע הפעולה מסתיים, הוא חוזר לprocess להמשך הרצה של הקוד.

בעזרת המנגנון האסינכרוני הזה, אפליקציה אחת של nodejs שרצה על thread אחד, יכולה לנהל אלפי בקשות במקביל – בלי הנטל של ניהול מקביליות וthreads, מה שמהווה בד"כ מקור לעומסים ולבאגים.

יתרונות

- **ביצועים גבוהים** - שימוש במנוע V8 JavaScript של גוגל, המהיר במיוחד בקימפול והרצת קוד. כמו כן, המנגנון האסינכרוני של nodejs המבוסס על event loop, מאפשר ריבוי משימות יעיל וביצועים גבוהים במשאבים נמוכים.
  כמו כן, הודות לריצה על process אחד, nodejs יעיל במיוחד בעבודה בעזרת microservices.
- **כתיבה בjavascript** – מפתחי frontend שמכירים את השפה יכולים לכתוב גם את צד השרת בקלות מבלי ללמוד שפה חדשה.
- **ספרית הרחבות גדולה** – ספריית ההרחבות של nodejs (NPM) מציעה מעל למליון חבילות open source שניתן להשתמש בהן לפיתוח מהיר ויעיל.

התקנה

1. גשו לאתר הרשמי של Node.js: https://nodejs.org
2. הורידו את הגרסה האחרונה המתאימה למערכת ההפעלה.
3. הריצו את קובץ ההתקנה.
4. בדקו שההתקנה הצליחה על ידי הרצת הפקודה הבאה בcommand line:

```
node -v
```

פקודה זו אמורה להחזיר את גרסת Node.js שהותקנה.

דוגמת קוד בסיסית

הדוגמא הבאה מכילה קטע קוד שיוצר שרת http בסיסי:

```
const { createServer } = require('node:http');

const hostname = '127.0.0.1';
const port = 3000;

const server = createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

הסבר הקוד:

```
const { createServer } = require('node:http');
```

טוען את הפונקציה createServer מתוך הספריה http (ספריה שכלולה בקוד הבסיסי של node)

```
createServer()
```

יוצר שרת HTTP שמאזין לבקשות.

```
 res.statusCode = 200;
 res.setHeader('Content-Type', 'text/plain');
 res.end('Hello World');
```

מחזיר תגובה עם סטטוס 200 וטקסט רגיל.

```
server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

מאזין לבקשות שיתקבלו ב3000 port ומעביר אותם לניהול הserver שנוצר.

הרצה

כדי להריץ את הקוד יש לשמור את הקובץ בסיומת **js** ולהריץ בcommand line:

```
node <filename>
```

לדוגמא:

```
node app.js
```

TypeScript

TypeScript היא שפה הבנויה על גבי javascript ומוסיפה לה types.

ניתן לקמפל בקלות את הקוד שנכתב בtypescript ולהמיר אותו לscript רגיל של javascript. בנוסף, ישנן ספריות להרצת typescript ישירות מבלי לקמפל קודם לקובץ javascript.

יתרונות:

- פיתוח נח ויעיל יותר.
- זיהוי באגים בזמן קומפילציה במקום בזמן ריצה.
- קוד קריא ומסודר.

במהלך הקורס כל הקוד שנכתוב יהיה בtypescript.

דוגמה לקוד TypeScript

```
function greet(name: string): string {
    return `Hello, ${name}!`;
}

console.log(greet('Node.js'));
```

הסבר הקוד:

```
function greet(name: string): string
```

פונקציה שמקבלת מחרוזת (name) ומחזירה מחרוזת.

```
return `Hello, ${name}!`
```

החזרת string שכולל בתוכו את השם שהתקבל בargument של הפונקציה.

```
console.log(greet('Node.js'))
```

הדפסת הערך המוחזר מהפונקציה לconsole.

הרצת קוד typescript

כדי להריץ typescript נשתמש בספריה ts-node המאפשרת הרצת קובץ typescript ישירות מהcommand line.

להתקנת החבילה יש להריץ את הפקודה:

```
npm i ts-node
```

להרצת הקובץ יש לשמור אותו בסיומת **ts**, ולהריץ:

```
npx ts-node <filename>
```

לדוגמא:

```
npx ts-node app.ts
```

דיבוג קוד typescript

כדי לדבג קוד typescript, נשתמש בקונפיגורציה של vscode שתקמפל תחילה את הקובץ ותמיר אותו לjs ואז תריץ את הjs שנוצר, תוך מיפוי הbreakpoints מקובץ הts לקובץ הjs.

יש ליצור קודם כל קובץ tsconfig.json שמכיל את הגדרות הקומפילציה:

```
{
    "compilerOptions": {
      "target": "ES5",
      "module": "CommonJS",
      "outDir": "out",
      "sourceMap": true
    }
}
```

כעת יש ליצור תיקיה בשם vscode. ותחתיה קובץ בשם launch.json שמכיל את הגדרות ההרצה:

```
{
    "version": "1.0.0",
    "configurations": [
        {
            "type": "node",
            "request": "launch",
            "name": "Launch Program",
            "program": "${workspaceFolder}/app.ts",
            "preLaunchTask": "tsc: build - tsconfig.json",
            "outFiles": ["${workspaceFolder}/out/**/*.js"]
        }
    ]
}
```

ניתן לשנות את app.ts לכל שם קובץ אחר כדי להריץ אותו.

לאחר יצירת הקבצים, אפשר להוסיף breakpoints בקובץ הts, ולהריץ את הקוד דרך Run and Debug.

את ההדפסות לconsole ניתן לראות בחלונית הDebug Console, דרכה גם אפשר להדפיס ערכים של משתנים בעת העצירה בbreakpoint.

Types

```
let age: number = 25;
let name: string = "John";
let isStudent: boolean = true;
```

Functions with types

```
function add(a: number, b: number): number {
    return a + b;
}
```

Interfaces

```
interface User {
    id: number;
    name: string;
}

const user: User = { id: 1, name: "Alice" };
```

Classes

```
class Person {
    constructor(public name: string, public age: number) {}
    greet() {
        return `Hello, my name is ${this.name}`;
    }
}

const person = new Person("David", 30);
console.log(person.greet());
```

Generics

```
function identity<T>(arg: T): T {
    return arg;
}

console.log(identity<number>(5));
console.log(identity<string>("Hello"));
```