NODE.JS FULLSTACK FEJLESZTÉS

5. forduló

A kategória támogatója: Emarsys

Ismertető a feladathoz

Event loop

Felhasznált idő: 00:00/15:00 **Elért pontszám:** 0/12

1. feladat 0/2 pont

Egy kéréseket kiszolgáló szervert szeretnénk készíteni az alábbiak szerint. **Mi a probléma a következő kóddal?**

```
import { createServer } from 'http';
import { on } from 'events';
import { setTimeout } from 'timers/promises';

const server = createServer().listen(8888);

for await (const [request, response] of on(server, 'request')) {
  await setTimeout(1000);
  response.end('ok');
}
```

Válasz

- ○ Túl későn iratkozik fel a request eseményre, emiatt elveszhetnek kérések.
- ● Egyszerre csak egy kérést tud feldolgozni.
  Ez a válasz helyes, de nem jelölted meg.
- ○ Sose fejeződik be a futása.

Magyarázat

Egyszerre csak egy kérést tud feldolgozni.

2. feladat 0/4 pont

A Node.js futtatásának lelke az Event loop. Melyik fut le biztosan az Event loop első fázisában?

Válasz

- [ ]
```
setImmediate(() => console.log('test'));
```
- [x]
```
setTimeout(() => console.log('test'), 1000);
```
Ez a válasz helyes, de nem jelölted meg.
- [ ]
```
socket.on('close', () => console.log('test'));
```
- [ ]
```
process.nextTick(() => console.log('test'));
```

Magyarázat

A **setImmediate** mindig a végrehajtandó callback-ek után fut, tehát az event loop közepén.

Habár a **setTimeout** nem garantált, hogy pontosan mikor fog lefutni, de az event loop szempontjából mindig a legelső fázisban fut (timers fázis).

A **process.nextTick** mindig az aktuális környezettől függ melyik fázisban fut le, amint befejeződött az őt indító feladat.

A **socket.on('close', ...)** futhat az event loop végén, amennyiben váratlanul befejeződik a socket futása.

3. feladat 0/6 pont

Mely állítás vagy állítások **igazak** a Node.js-ben használt require és import modul betöltést szolgáló utasításokra?

Válaszok

- [ ] Mindkettő az ECMAScript szabvány része.
- [ ] Ekvivalensen használhatóak, ugyanúgy töltik be a megadott modult.
- [x] Egy fájlon belül nem használhatóak egyszerre.

Ez a válasz helyes, de nem jelölted meg.
- [x] A require-höz hasonlóan az import is tud függvényszerűen működni. (import('test.mjs'))

Ez a válasz helyes, de nem jelölted meg.

Magyarázat

Mindkettő az ECMAScript szabvány része: nem igaz, a require nem resze.

Ekvivalensen használhatóak, ugyanúgy töltik be a megadott modult: nem igaz, másképp viselkedik a kettő. Az import aszinkron tölti be a háttérben a megadott modult.

A require nem használható ESM környezetben.

Az import('test.mjs') habár nem függvény, használatát tekintve hasonlóan működik és dinamikus module betöltésre használható.