Devoir Maison - Semaine 2

Les promises et la syntaxe async/await

Lecture suggérée :

- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Using_promises
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise

Pourquoi et dans quels cas est-il considéré comme préférable d'utiliser une promise, plutôt que de bloquer sur un appel IO, alors que le code est plus compliqué à comprendre ?

Comment ré-écrire le code suivant avec des callbacks ? Avec des promises ? Avec la syntaxe async/await ? Attention, les fonctions `f1, f2, f3, f4` sont bloquantes, ce qu'on veut éviter.

```
const r1 = f1()
const r2 = f2(r1)
const r3 = f3(r1)
const r4 = f4(r2, r3)
console.log(r4)
```

Qu'appelle-t-on un « sucre syntaxique » ? Quel sucre syntaxique avez-vous utilisé sur l'exercice précédent ?

Pourquoi utiliser des promises plutôt que des callbacks ?

Dans quelle mesure les promises sont-elles utiles pour réaliser des systèmes « réactifs » (comme définis dans le « Reactive Manifesto ») ?

Les migrations de données

Quel(s) outil(s) permettent de gérer l'accès à la base de donnée et le mapping entre les objets du domaine et les tables stockées en base ?

Quand on veut faire évoluer la structure de la base de données (par exemple, rajouter un champ), il est inenvisageable de supprimer les données et de recréer la base : il faut modifier les données en place. Cette opération se nomme une migration. Quelle solution pouvez-vous mettre en place pour gérer les migrations de données ?

Comment tester les migrations ?

Design patterns

Qu’est-ce qu’un design pattern ? Citer les ouvrages qui vous semblent faire référence.

Quels sont leurs avantages et leurs limitations ?

Décrire brièvement 3 design patterns de votre choix.

Domain-Driven Design

Lire cette introduction à Domain-Driven Design
https://khalilstemmler.com/articles/domain-driven-design-intro/