

Qualité de développement

Tests unitaires en TypeScript

Bonnes pratiques

Il est indispensable de séparer les fichiers contenant le code, des fichiers contenant les tests. Vous devez créer un répertoire `src` pour y placer vos fichiers de code et un répertoire `tests` pour y placer vos fichiers de tests.

Exercice 1

Corriger la fonction `multAdd` pour que l'ensemble des tests passent avec succès.

Exercice 2

Dans le fichier `MesMaths.ts`, ajouter la fonction suivante :

```
export function puissanceMult(x: number, y: number): number {
  let val = 1;

  for (let i = 0; i < y; i++) {
    val *= x;
  }

  return val;
}
```

On considère dans un premier temps que les paramètres sont nécessairement des entiers.

1. Quels sont les cas à tester ? Pour chacun donner le résultat attendu.
2. Ajouter les tests correspondants dans le fichier `puissanceMult.test.ts`.
3. Lancer les tests avec `deno test`.
4. Lancer les tests avec `deno test tests/puissanceMult.test.ts`.
5. Quelle est la différence ?
6. Quel cas pose problème ?
7. Modifier la fonction `puissanceMult` pour qu'elle lève une exception définie ci-dessous avec un message adéquat lorsque ce cas se présente.

```
export class ErreurPuissanceMult extends Error {
  constructor(msg: string) {
```

```
    super(msg);  
    this.name = "ErreurPuissanceMult";  
  }  
}
```

8. Modifier votre fichier de test pour qu'il prenne en compte cette nouvelle exception. Vous devez utiliser `assertThrows`.
9. Traiter le cas dans la fonction `puissanceMult` avec les tests adéquats pour vérifier que le second paramètre est bien un entier. Vous pouvez utiliser la fonction `Number.isInteger` pour cela :

```
let x = 2;  
let y = 3.14;  
  
console.log("Number.isInteger(x) =", Number.isInteger(x));  
console.log("Number.isInteger(y) =", Number.isInteger(y));
```

```
Number.isInteger(x) = true  
Number.isInteger(y) = false
```

Exercice 3

Tester la fonction `creerInitiales` de l'exercice 1 du TD sur les exceptions.

Exercice 4

Tester la méthode `setINE` de l'exercice 2 du TD sur les exceptions. Vous devrez notamment tester les cas où l'INE est invalide et s'il est valide que l'objet contient la bonne valeur (vous pouvez ajouter un getter pour cela).

Exercice 5

Tester la fonction de l'exercice 3 du TD sur les exceptions (les dates au format européen et US).