SAé 2.01 - Partie Tests et Qualité

La partie "Qualité de développement" de cette saé sera évaluée de la façon suivante :

1. tous les objets métier du modèle (partie M de l'architecture MVC) devront sécuriser leurs entrées, en vérifiant les paramètres donnés aux constructeurs et/ou aux setters (à vous d'optimiser cela en vous interdisant toute duplication de code). En cas de non conformité, une exception doit être lancée. La liste des règles de gestion à respecter est indiquée ci-dessous.
2. le langage support de cette saé étant TypeScript, les données doivent être correctement typées. Si toutes les données sont de type `String`, cela n'a aucun intérêt d'utiliser TS.
3. le code doit être bien structuré, bien organisé, sans duplication et avec des méthodes courtes.
4. tous ces objets métier devront être testés en utilisant "deno test".

Exemple de ce qui est attendu :

1. objets métier du modèle MVC

classe : abonnement

méthode : set dateAbonnement

la date passée en paramètre de la méthode doit être une date passée ou du jour. Votre méthode doit tester le paramètre en entrée : si celui-ci est conforme à ce qui précède, l'attribut d'instance est modifié ; sinon, une exception avec un message parlant doit être lancée, par exemple "La date d'abonnement est située dans le futur, ce n'est pas possible !"

2. test deno :

test 1 : la méthode set dateAbonnement est appelée avec un paramètre valide (`new Date("2024-7-10")`), une méthode de deno est utilisée pour vérifier que le getter de dateAbonnement renvoie bien une instance de Date contenant cette date après l'appel du setter

test 2 : la méthode set dateAbonnement est appelée avec une valeur "limite" (la date du jour : `new Date()`), une méthode de deno est utilisée pour vérifier que le getter de dateAbonnement renvoie bien une instance de Date contenant la date du jour après l'appel du setter

test 3 : la méthode set dateAbonnement est appelée avec un paramètre invalide (date du jour + 2 jours), une méthode de deno est utilisée pour vérifier que le setter lance bien une exception

Remarque 1 : la partie SQL des objets métier n'est pas à tester !

Remarque 2 : si les imports ou différents exports de vos classes métier vous posent problème lors des tests deno, vous pouvez tout à fait fournir vos classes métier en supprimant les parties SQL et imports/exports

qui vous gênent

Liste des règles de gestion à respecter :

- Abonnement :

  abon_num : un entier supérieur à 0

  abon_date : une date inférieure ou égale à la date du jour

  abon_comment : peut être vide, mais de longueur maximale 400 caractères si renseigné

  adh_num : un entier supérieur à 0

- Adhérent :

  adh_num : un entier supérieur à 0

  adh_civ : valeurs acceptées "M." ou "Mme"

  adh_nom, adh_prenom : longueur comprise entre 2 et 20 caractères

  adh_adr : peut être vide, mais limitée à 50 caractères si renseignée

  adh_cp : peut être vide, mais limité à 5 caractères si renseigné

  adh_ville : peut être vide, mais limitée à 30 caractères si renseignée

  adh_mel : peut être vide, mais limité à 50 caractères si renseigné ; doit respecter une expression régulière vérifiant les emails

  csp_num : une chaîne comprise entre 1 et 5 caractères (conformément à la base de données)

- CSP :

  csp_num : une chaîne comprise entre 1 et 5 caractères

  csp_lib : une chaîne non vide limitée à 100 caractères

- Thème :

  theme_num : un entier supérieur à 0

  theme_lib : une chaîne entre 2 et 20 caractères

  theme_tarif : un réel supérieur à 0

- Adhésion :

  abon_num, theme_num : voir les classes Abonnement et Thème