Test

Paul Boulesteix - 20198839
Richard Lao - 20278343

October 8, 2024

1 W3CColorNamesTest

Raison : Nous testons différentes valeurs ainsi que des valeurs inattendues dans cette classe. Nous avons choisi cette fonction parce que l'aspect couleur du programme est assez essentiel, en particulier sa capacité à reconnaître les couleurs, pour ainsi dire. De plus, la gestion précise des couleurs est cruciale pour l'expérience utilisateur et l'affichage correct des éléments graphiques dans l'application. Assurer la fiabilité de cette fonctionnalité permet d'éviter des erreurs visuelles qui pourraient compromettre l'interface utilisateur.

Chemin : `src\test\java\com\marginallyclever\convenience\W3CColorNamesTest.java`

- `testGetKnownColorName` : Teste que la méthode `get` retourne l'objet `Color` correct pour un nom de couleur connu.
- `testGetUnknownColorName` : Teste que la méthode `get` retourne `null` pour un nom de couleur inconnu.
- `testGetKnownColorObject` : Teste que la méthode `get` retourne le nom de couleur correct pour un objet `Color` connu.
- `testGetUnknownColorObject` : Teste que la méthode `get` retourne `null` pour un objet `Color` inconnu.

2 SelectSpinnerTest

Raison : Nous testons cette classe car elle fait partie des classes select , qui sont essentielles pour contrôler le programme lui-même et ses entrées. S'assurer que les méthodes Input/Select fonctionnent correctement peut éviter de nombreux problèmes lors des tests ultérieurs. De plus, les composants de sélection tels que les spinners sont fréquemment utilisés dans l'interface utilisateur pour permettre aux utilisateurs de choisir des options de manière intuitive. Garantir leur bon fonctionnement est donc primordial pour une expérience utilisateur fluide et sans erreurs.

Chemin : `src\test\java\com\marginallyclever\makelangelo\select\SelectSpinnerTest.java`

- `getValue initialValue returnsCorrectValue` : Teste que la valeur initiale du spinner est retournée correctement.
- `setValue validValue updatesValue` : Teste que le réglage d'une valeur valide met à jour la valeur du spinner.
- `setValue valueOutOfRange throwsIllegalArgumentException` : Teste que le réglage d'une valeur hors de la plage autorisée lance une `IllegalArgumentException`. Ce test est réalisé deux fois, avec des valeurs hors limite des deux côtés :
  - `setValue valueBelowRange throwsIllegalArgumentException` : Teste que le réglage d'une valeur en dessous de la plage autorisée lance une `IllegalArgumentException`.
  - `setValue valueAboveRange throwsIllegalArgumentException` : Teste que le réglage d'une valeur au-dessus de la plage autorisée lance une `IllegalArgumentException`.
- `getValue afterSettingValue returnsUpdatedValue` : Teste que la valeur retournée après avoir réglé une nouvelle valeur est la valeur mise à jour.

3 MathHelper

Raison : Nous testons la classe `MathHelper` car elle contient des fonctions utilitaires mathématiques essentielles utilisées dans diverses parties de l'application. Assurer la précision et la fiabilité de ces méthodes est crucial pour la validité des calculs géométriques et autres opérations mathématiques effectuées par le programme. En validant ces fonctions, nous évitons la propagation d'erreurs qui pourraient entraîner des dysfonctionnements importants dans d'autres modules du système.

Chemin : `test\java\com\marginallyclever\convenience\helpe`

- `testLerpDouble` : Le test `lerp` effectue une interpolation linéaire entre deux nombres en fonction d'un paramètre t.

4 TestStringHelper

Raison : Nous testons la classe `TestStringHelper` car elle fournit des fonctions utilitaires essentielles pour la manipulation des chaînes de caractères dans l'application. Garantir le bon fonctionnement de ces méthodes assure une gestion correcte des formats de données textuelles, ce qui est important pour l'affichage, le traitement des entrées utilisateurs, et l'intégrité des données textuelles manipulées par le programme.

Chemin : `test\java\com\marginallyclever\convenience`

- `testGetElapsedTime` : Le test `getElapsedTime` convertit un nombre de secondes en une chaîne de caractères au format "HH:MM:SS". .

5 TestColorRGB(javaFaker)

Raison : Nous testons la classe `ColorRGB` car elle gère les représentations de couleurs RGB, essentielles pour le rendu graphique de l'application. Assurer la précision des opérations sur les couleurs, telles que l'addition, la soustraction et la conversion en entier ou en chaîne hexadécimale, est crucial pour maintenir la cohérence visuelle et éviter les erreurs de couleur qui pourraient nuire à l'expérience utilisateur.

Chemin : `test\java\com\marginallyclever\convenience\TestColorRGB.java`

- `testWithFaker` : Teste que le constructeur de `ColorRGB` initialise correctement les composantes rouge, verte et bleue avec des valeurs aléatoires générées par `Faker`.

6 Coverage

La couverture des tests a augmenté de 24,4% à 24,6%. Cela représente une amélioration légère mais notable grâce à l'ajout de nouveaux tests pour les classes `W3CColorNamesTest`, `SelectSpinnerTest`, `MathHelper`, et `TestStringHelper`.