Université Paul Sabatier – Toulouse III
IUT A - Toulouse Rangueil
Projet tuteuré #20

Antoine de Roquemaurel
Mathieu Soum
Geoffroy Subias
Marie-Ly Tang
Groupe B

Pour Monsieur Max Chevalier (Resonsable projet)
Madame Caroline Kross (Tutrice)
Monsieur Thierry Millan (Client)

Les plans et les cahiers de tests

Bibliothèque d'objets graphiques UML

Toulouse, le 22 janvier 2012

Avant-propos

Ce document fais partis d'un groupe de documents permettant de montrer notre approche pour développer une bibliothèque d'objets graphique UML [1]. UML est un langage permettant de modéliser une application, de la concevoir.

Présentation du groupe

Notre groupe projet est composé de quatre étudiants de deuxième année de DUT Informatique à l'IUT [2] A de Toulouse, voici la composition de l'équipe :

- Antoine de Roquemaurel
- Mathieu Soum
- Geoffroy Subias
- Marie-Ly Tang

Nous avons monté ce groupe, car nos compétences sont complémentaires et que nous savons déjà comment chacun travaille.

Antoine de Roquemaurel et Mathieu Soum sont spécialisés en programmation par objet, Geoffroy Subias est le plus compétent lorsqu'il s'agit de modélisation UML [3] et Marie-Ly Tang s'occupera principalement de l'organisation et de la gestion de projet.

Nos compétences sont différentes mais sont complémentaires pour mener à bien notre projet.

Fonctionnement du document

Ce document est un document expliquant notre approche pour développer une bibliothèque d'objets graphiques UML.

Dans ce dossier, vous pourrez repérer diverse notations, cette introduction à pour but de vous expliquer les notations afin que vous puissiez lire en toute sérénité.

Le glossaire

Un mot dans le glossaire à une police particulière, vous pourrez savoir qu'un mot est dans le glossaire lorsque vous repérerez un mot avec la police suivante : leMotDansLeGlossiare. Si vous voyez cette police, vous pouvez donc vous référez à l'annexe **??** page **??**.

1. Unified Modelling Language
2. Institut Universitaire de Technologie
3. Unified Modelling Language

Les noms de méthode, d'attribut ou de classe

Les mots se référant à un nom présent dans du code ont une police particulière, une police type "machine à écrire", si vous voyez la police suivante, c'est que c'est un nom de méthode, d'attribut ou de classe : `uneFonction`.

Les noms de paquetage

Les noms de paquetages utiliseront une police particulière, afin que l'on puisse les différencier d'une classe ou d'une méthode, il seront écris comme ceci : unPaquetage.

Les notes de bas de page

Nous utilisons régulièrement des notes de bas de pages, pour donner un acronyme, pour expliquer plus en détail une notion, ces notes de bas de pages sont un numéro en exposant, vous trouverez la note correspondante en bas de la page courante, comme ceci [4].

Les liens hypertext

Dans le document, nous pouvons faire référence à un lien d'un site web, tous les liens seront donc symbolisés par une petite puce, et une police particulière comme ceci :
▷ `http://monLien.fr/index.html`

4. Ceci est une note de bas de page

Table des matières

Chapitre 1

Les plans de tests

1.1 Les plans de non régression

Notre projet est de développé une bibliothèque UML [1], aucun système n'existant au départ, il est donc inutile de faire un test de non régression étant donné que nous ne continuons pas un système, il n'y aura pas de régression.

1.2 Les tests de validation

Nous avons développé un démonstrateur qui utilise notre bibliothèque et qui fait office de test de validation. Il nous permet de vérifier que toutes les attentes du client sont respectés.
Avec ce démonstrateur, nous pouvons montrer au client les différentes possibilités de la bibliothèque, ainsi qu'un exemple d'utilisation.

1.2.1 Jeux d'essais

1.2.1.1 Dessiner un diagramme complet

Nous avons effectués plusieurs jeux d'essais, le but est de pouvoir dessiner pour chacun des types de diagrammes demandés par le client, un diagramme utilisant tous les composant de la norme UML 2.0 et respectant cette même norme, ceci pour chacun des diagrammes demandés par notre client.

- Dessiner un diagramme de classe.
- Dessiner un diagramme de séquence.
- Dessiner un diagramme de cas d'utilisation.

1.2.1.2 Respecter les contraintes d'ajout d'éléments

Pour chacun des diagrammes, les contraintes doivent être respectés, ainsi en fonction du type de diagramme, certains éléments ne doivent pas pouvoir être inséré. Voici la liste des éléments autorisés en fonction du type de diagramme.

Diagramme de cas d'utilisation
- Cas d'utilisation
- Acteur actif

Diagramme de classe
- Classe
- Interface

Diagramme de séquence
- Traitement
- Acteur actif
- Acteur passif
- Ligne de vie

1. Unified Modeling Language

1.2.1.3 Respecter les contraintes de liaisons d'éléments

Nous devons également respecter les contraintes de liaison entre les différents éléments. Ainsi voici la liste des liaisons possible.

Diagramme de classe
- Entre deux classes
  - Agrégation
  - Composition
  - Association
  - Spécialisation
- Entre une classe et une interface
  - Spécialisation
  - Dépendance fonctionnelle

Diagramme de cas d'utilisation
- Entre un acteur un cas d'utilisation
  - Association
- Entre deux cas d'utilisations
  - Dépendance fonctionnelle

Diagramme de séquence
- Entre deux traitements
  - Dépendance fonctionnelle
  - Déclenchements

1.2.2 Résultats obtenus

Nous vérifierons que les diagrammes sont bien réalisés, qu'il n'y a aucune erreur, et qu'ils correspondent aux besoins du client.
Également, nous vérifierons que nous ne pouvons pas relier ou ajouter des éléments non autorisés par la norme UML 2.0.

1.3 Les tests d'intégration

Nous avons développé un démonstrateur qui utilise notre bibliothèque et qui fait office de test d'intégration démontrant que notre système est utilisable en tant que composant par un logiciel extérieur. Il nous permet également de montrer au client les possibilités de notre bibliothèque.

1.3.1 Les composants à tester

Notre bibliothèque d'objet graphique en tant que composant de notre démonstrateur.

1.3.2 Les jeux d'essais

Dessiner un diagramme de classe, un diagramme de séquence ainsi qu'un diagramme de cas d'utilisation chacun utilisant tous les objets graphiques possibles.
Une fois ces diagrammes dessinés, nous testerons la suppression [2], l'ajout de méthodes et d'attributs dans une classe, ainsi que la modification de données.

1.3.3 Les résultats attendus

Représentation graphique des différents diagrammes respectant les normes UML 2.0 avec possibilité d'ajout ou de suppression de données.

2. La suppression d'un élément devant supprimer les liens de l'élément supprimé

1.4 Les tests unitaires

Les tests unitaires servent à tester chacune des méthodes de notre bibliothèque.
Nous avons utiliser JUnit pour tester toutes les méthodes de notre bibliothèque. Les tests sont validés si la bibliothèque `JUnit` nous indique que tous les tests passent et ne posent pas de problème. La description des tests unitaires est disponible sous format Javadoc à l'annexe A page 7. Elle est également disponible sous format HTML à l'adresse suivante :
▷ `http://documentation.joohoo.fr/libUML/junitTests/index.html`

1.5 Le test de charges

1.5.1 Le jeux d'essai

Le test de charge permet de savoir si la création d'un grand nombre d'élément graphique ne posera pas de problème. Ainsi, nous allons créer les éléments suivants dans un diagramme sans contraintes.

- 20 Classes
- 20 Cas d'utilisation
- 20 Traitements
- 20 Acteurs actifs
- 20 Acteurs passifs
- 20 interface
- 60 liens

Une fois ces éléments suivants créer, nous supprimerons 40 éléments, qui devront supprimer les liens reliés aux éléments.

1.5.2 Les résultats attendus

Aucune exception, les éléments ont été bien créer et certains éléments ont été supprimés avec les liens.

Chapitre 2

Les cahiers de tests

2.1 Création du démonstrateur

Pour les tests de validation, nous avons donc créer un démonstrateur, ce démonstrateur à évolué tout au long du développement de la bibliothèque UML, lorsque nous rajoutions des éléments dans la bibliothèque, nous modifions le démonstrateur pour pouvoir les intégrer.

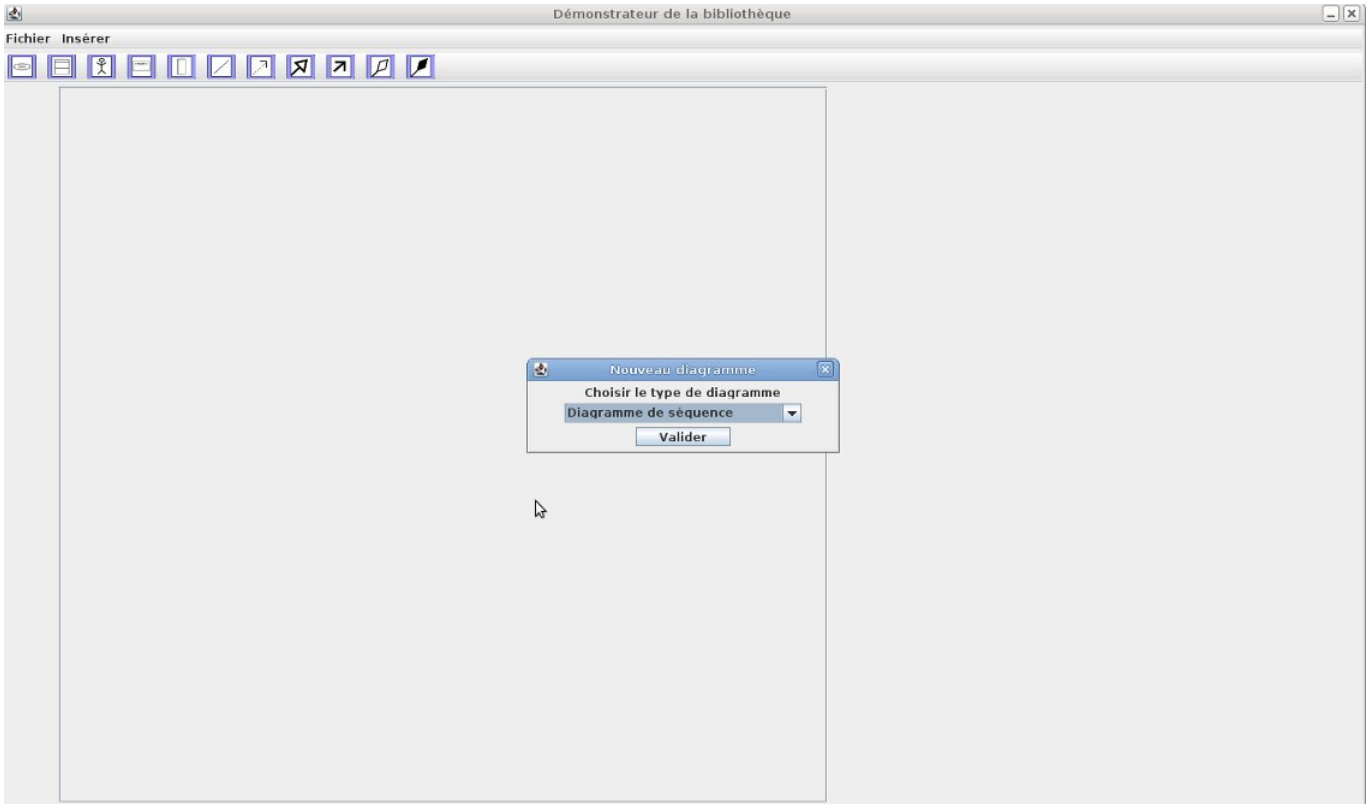


FIGURE 2.1 – Ouverture du démonstrateur (Choix du type de diagramme)

2.2 Les tests de validation

2.2.1 Dessiner un diagramme de cas d'utilisation

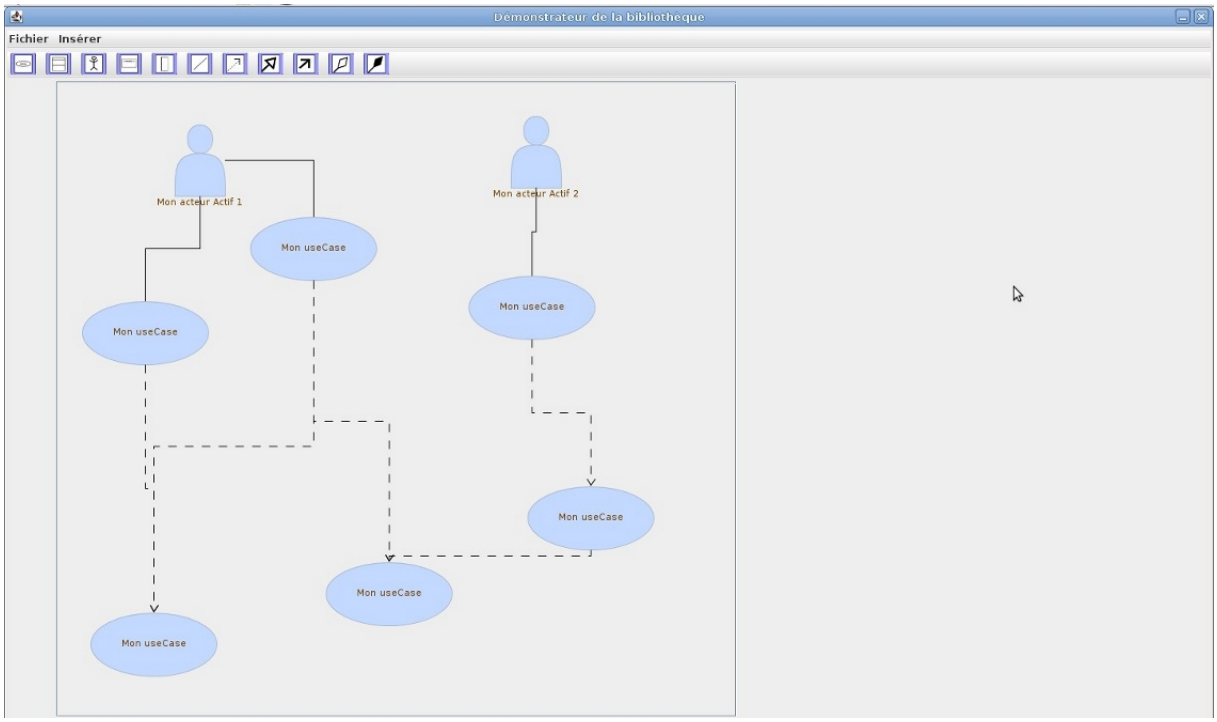


FIGURE 2.2 – Création d'un diagramme de cas d'utilisation simple

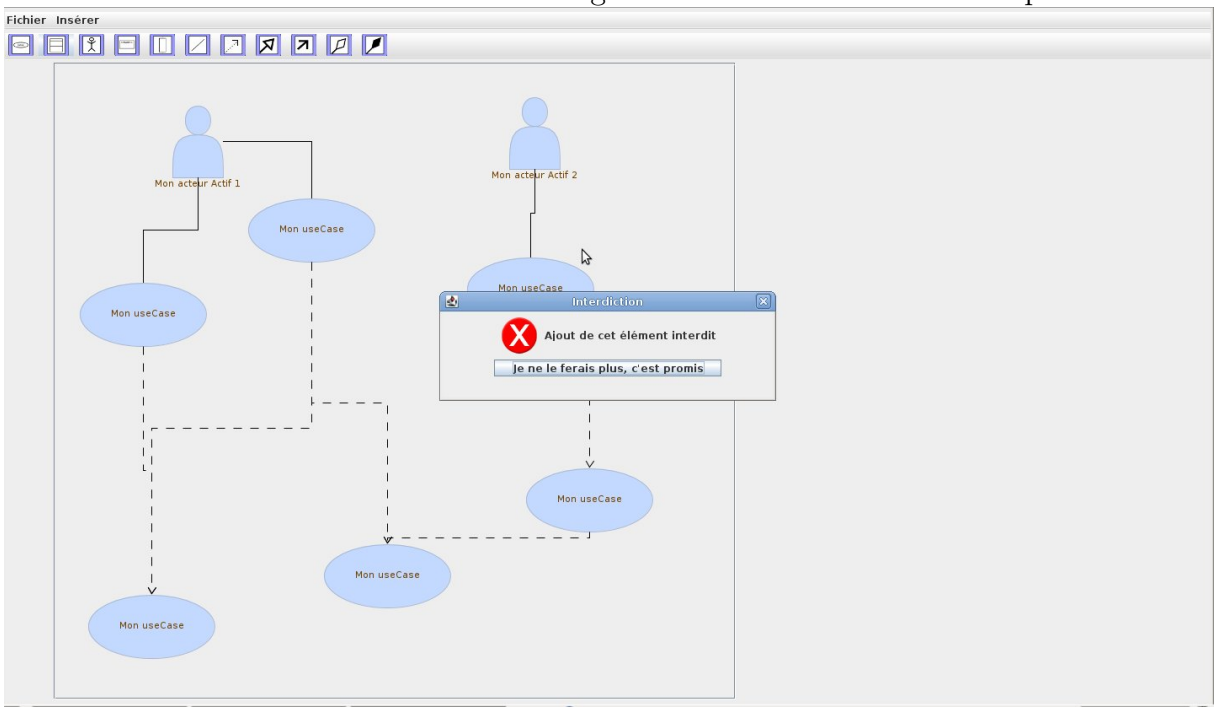


FIGURE 2.3 – Validation de l'interdiction d'ajouter des éléments non autorisés

2.2.2 Dessiner un diagramme de classe

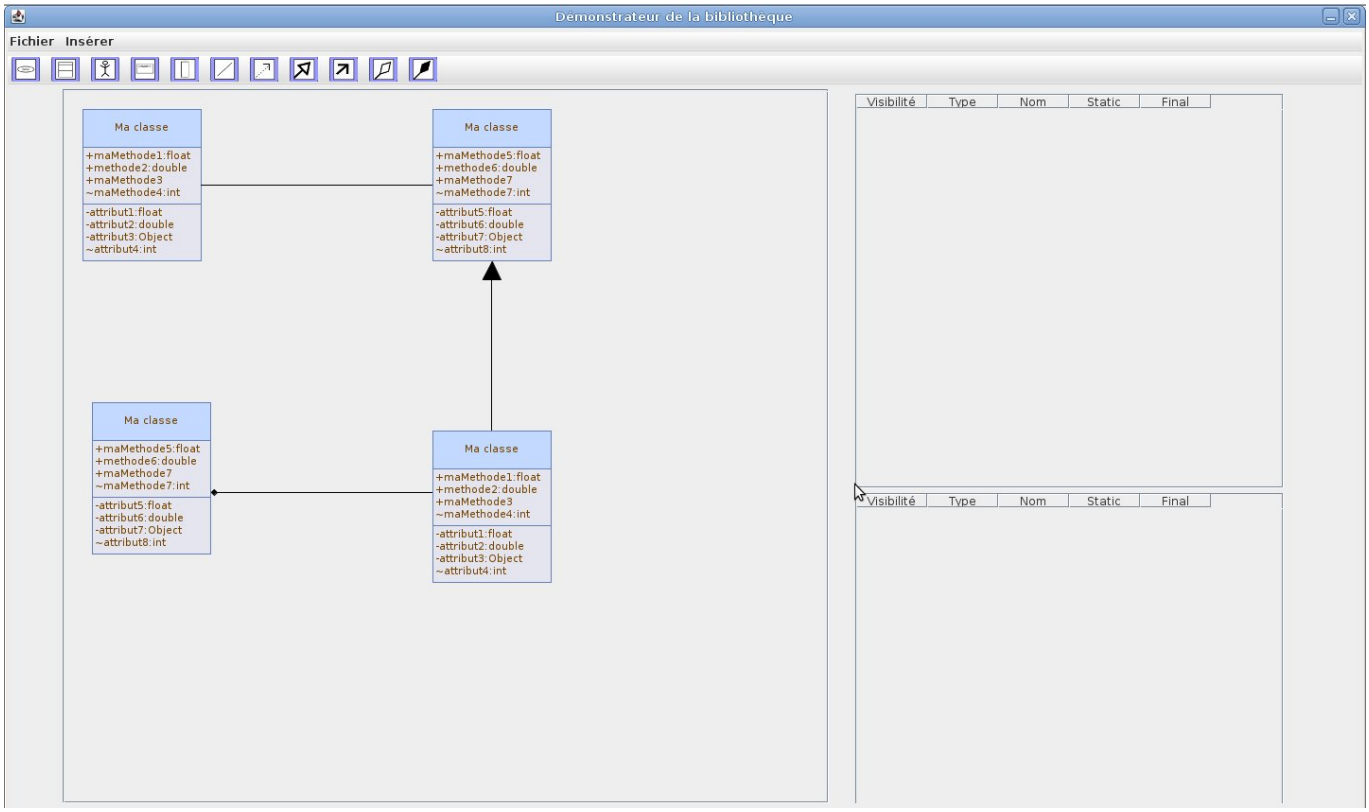


FIGURE 2.4 – Création d'un diagramme de classe simple

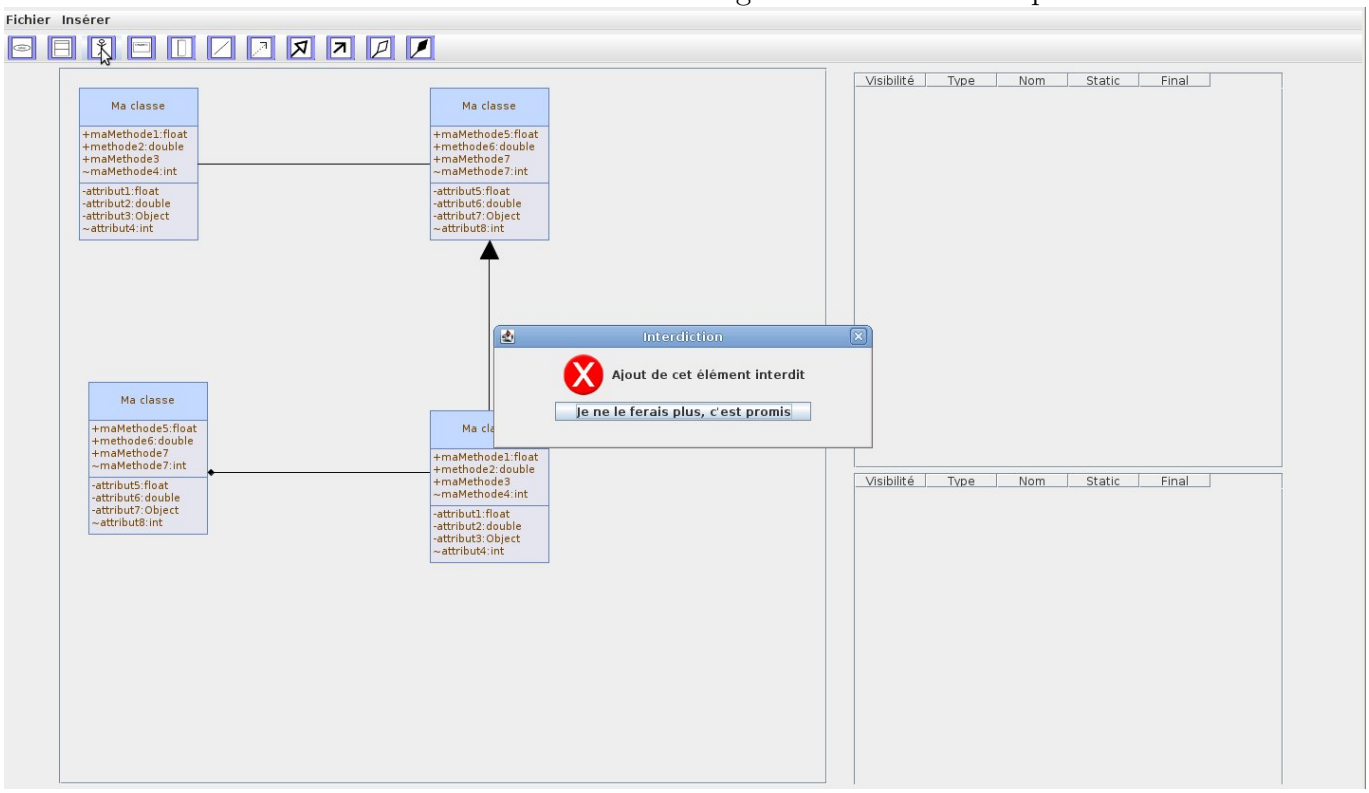


FIGURE 2.5 – Validation de l'interdiction d'ajouter des éléments non autorisés

2.2.3 Dessiner un diagramme de séquence

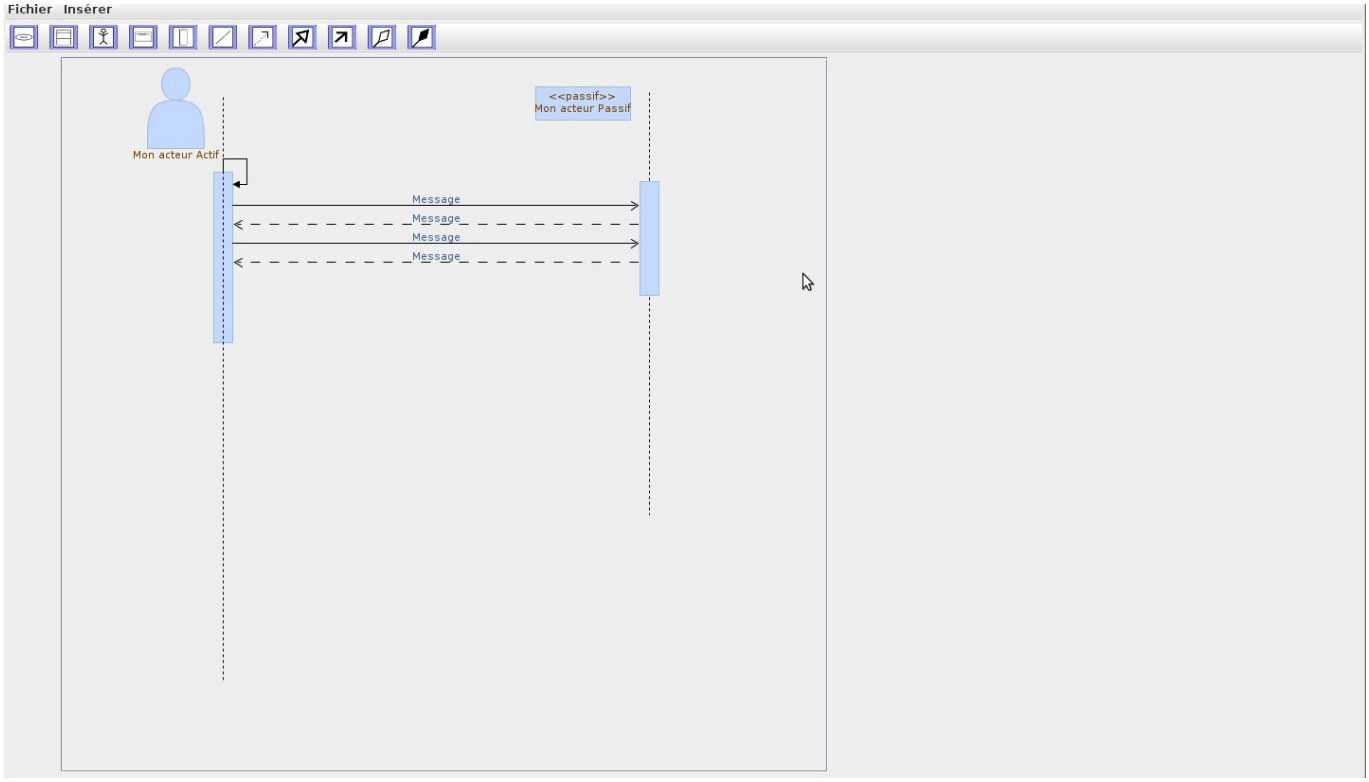


FIGURE 2.6 – Création d'un diagramme de séquence simple

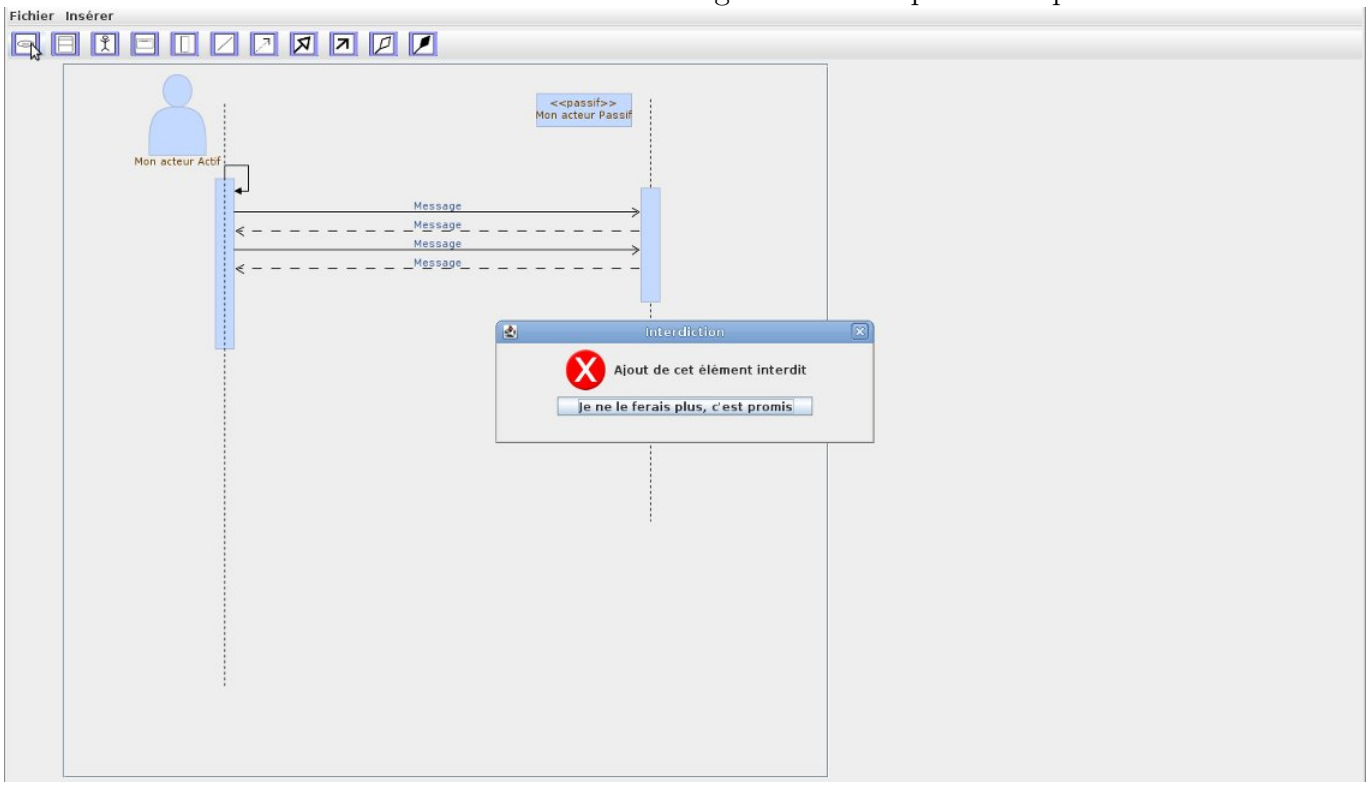


FIGURE 2.7 – Validation de l'interdiction d'ajouter des éléments non autorisés

2.3 Les tests d'intégration

Nous avons créer un diagramme sans aucune contrainte, tous les éléments pouvant donc être reliés entre eux.

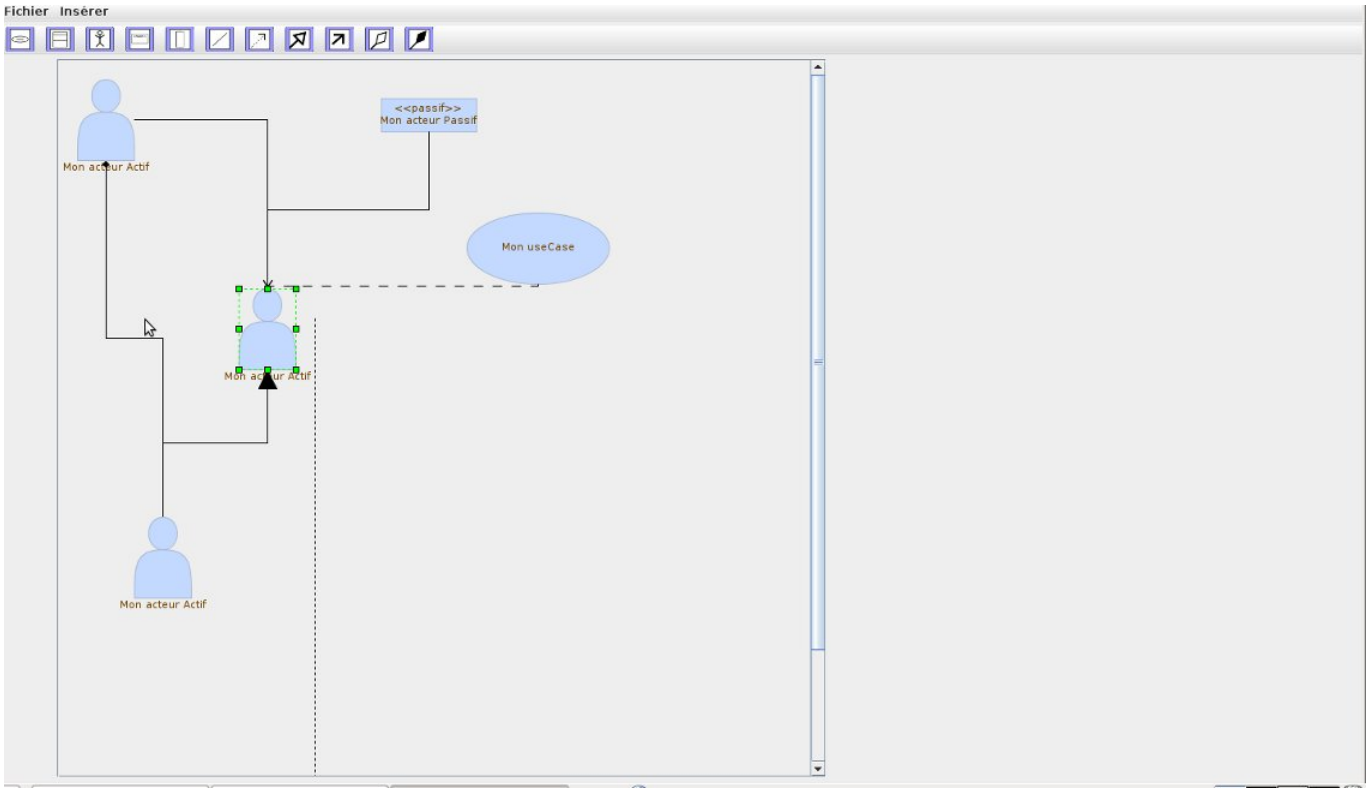


FIGURE 2.8 – Diagramme sans aucune contrainte

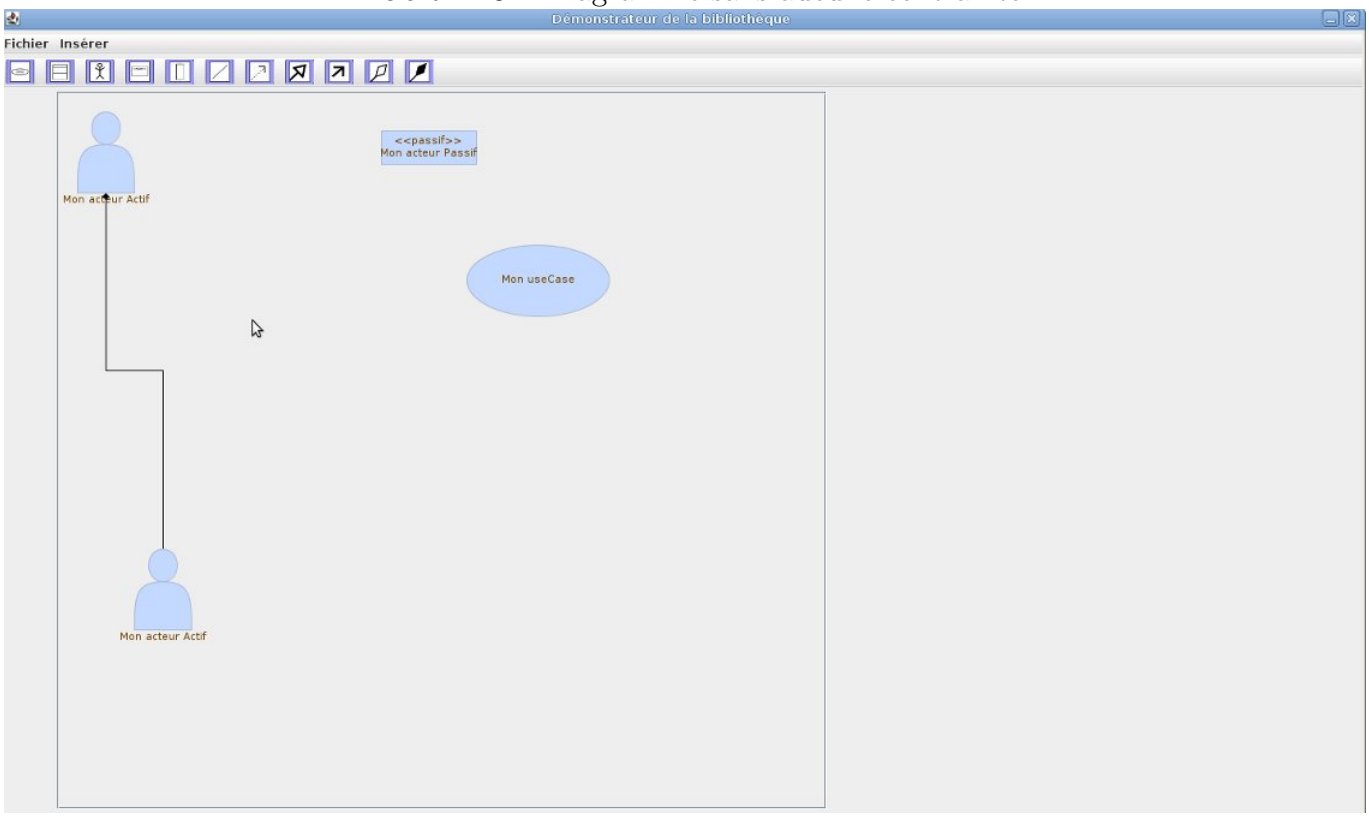


FIGURE 2.9 – Le même diagramme après suppresion de l'acteur au centre

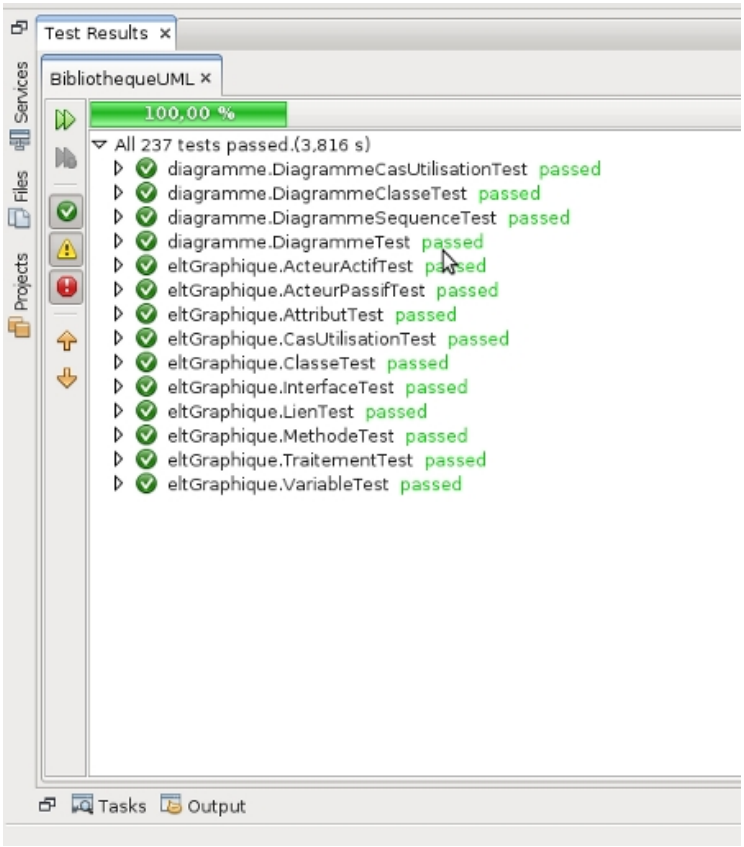
2.4 Les tests unitaires

2.4.1 Création des tests

Nous avons donc créer des tests unitaires `JUnit`, une classe de test pour une classe à tester. Chaque méthodes doivent être testés, avec un ou plusieurs tests.
Vous pouvez avoir les détails des tests unitaires avec la documentation en Annexe A page 7.

2.4.2 Résultat des tests

Après lancement des tests unitaires, ils sont tous passéss. Nous pouvons donc en conclure que les différents modules de la bibliothèque UML fonctionnent.



2.5 Le test de charge

2.5.1 Création du test

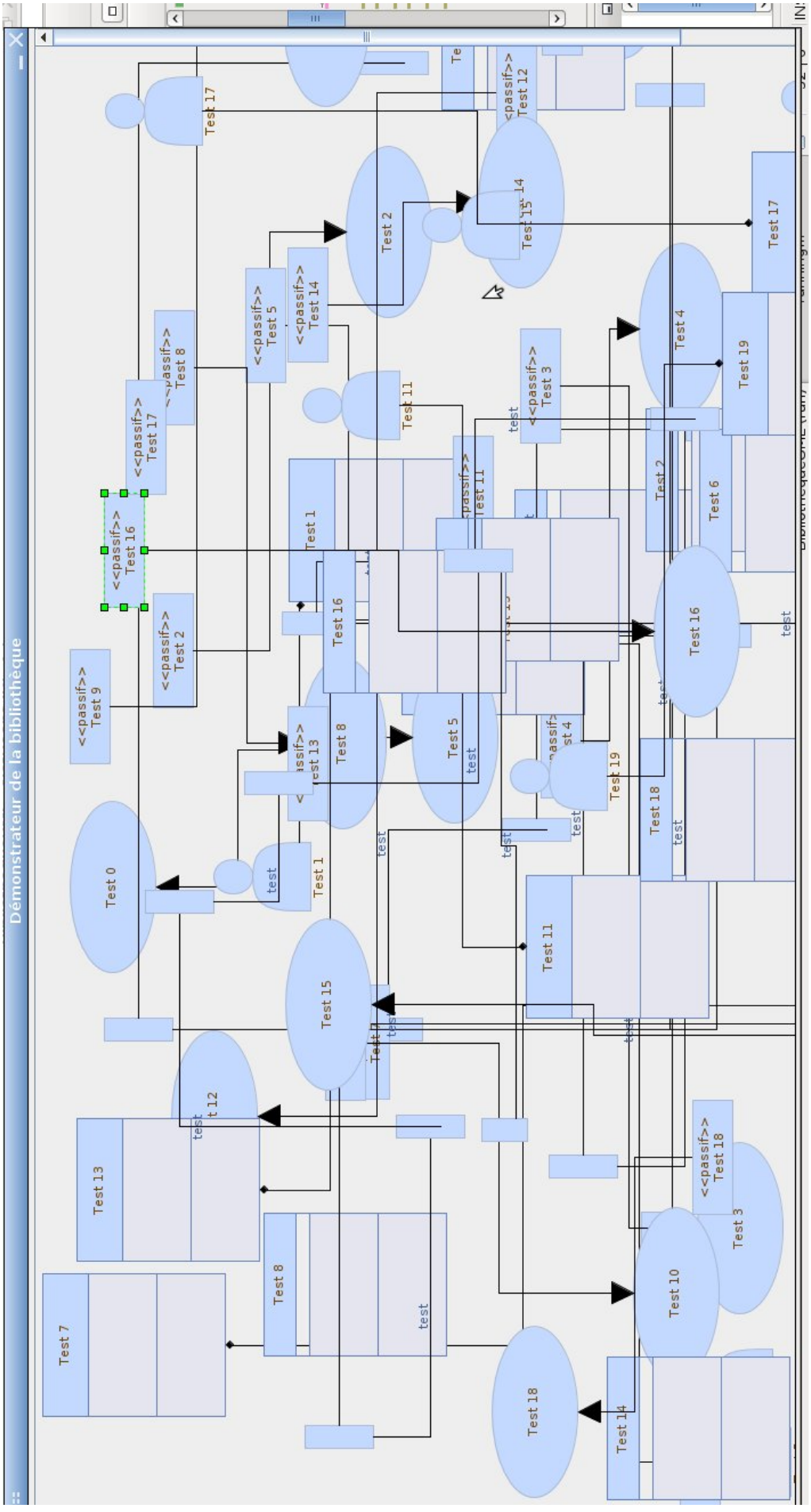
Le test de charge à été effectuée en créant les éléments graphiques répértorié dans les plans de tests. Voici le code permettant d'effectuer ce tests :

```
1  public static void main(String[] args){
2    int valeurMinX = 0, valeurMaxX = 1300;
3    int valeurMinY = 0, valeurMaxY = 700;
4    Random r;
5    Position position;
6
7    Classe classes[] = new Classe[20];
8    CasUtilisation casUtilisations[] = new CasUtilisation[20];
9    Traitement traitements[] = new Traitement[20];
10   Acteur acteursActifs[] = new ActeurActif[20];
11   Acteur acteursPassifs[] = new ActeurPassif[20];
12   Lien liens[] = new Lien[60];
13
14   ChargeTest fenetre = new ChargeTest();
15   fenetre.afficherFenetre();
16
17   /* créer des éléments */
18   for(int i = 0; i < 20 ; ++i) {
19     /* on instancie une classe */
20     position = calculPositionAleatoire();
21     classes[i] = new Classe(fenetre.getPanneauGraph().getGraph(),
22         new Diagramme(), "Test "+i, position);
23
24     /* on instancie un cas d'utilisation */
25     position = calculPositionAleatoire();
26     casUtilisations[i] = new CasUtilisation(
27         fenetre.getPanneauGraph().getGraph(),
28         new Diagramme(), "Test "+i, position);
29
30     /* on instancie un traitement */
31     position = calculPositionAleatoire();
32     traitements[i] = new Traitement(
33         fenetre.getPanneauGraph().getGraph(),
34         new Diagramme(), "Test "+i, null, position, false);
35
36     /* on instancie un acteur passif*/
37     position = calculPositionAleatoire();
38     acteursPassifs[i] = new ActeurPassif(
39         fenetre.getPanneauGraph().getGraph(),
40         new Diagramme(), "Test "+i, position);
41
42     /* on instancie un acteur actif*/
43     position = calculPositionAleatoire();
44     acteursActifs[i] = new ActeurActif(
45         fenetre.getPanneauGraph().getGraph(),
```

```
46          new Diagramme(), "Test "+i, position);
47
48      /* on créer tous les éléments instanciés précédemment */
49      classes[i].creer();
50      casUtilisations[i].creer();
51      traitements[i].creer();
52      acteursPassifs[i].creer();
53      acteursActifs[i].creer();
54
55      /* on relie les éléménts créer */
56      if(i != 0) {
57          traitements[i].ajouterMessage(traitements[i-1], "test",
              TypeLien.ASSOCIATION);
58      }
59      liens[i+1] = new Lien(acteursActifs[i], classes[i], fenetre.
          getPanneauGraph().getGraph(), new Diagramme(),
60          TypeLien.AGREGATION);
61      liens[i+2] = new Lien(acteursPassifs[i], casUtilisations[i],
          fenetre.getPanneauGraph().getGraph(),
62          new Diagramme(), TypeLien.SPECIALISATION);
63
64      /* on affiche les liens */
65      liens[i+1].creer();
66      liens[i+2].creer();
67    }
68
69    /* on supprime une vingtaine d'éléments */
70    for(int i =0 ; i < 20 ; i++){
71      if(i % 2 == 0){
72        acteursActifs[i].supprimer();
73      } else if(i % 3 == 0) {
74        classes[i].supprimer();
75      } else if(i% 5 == 0){
76        traitements[i].supprimer();
77      } else {
78        casUtilisations[i].supprimer();
79      }
80    }
81  }
```

Listing 2.1 – Code du test de charge

2.5.2 Résultat du test



Annexe A

Glossaire

Bibliothèque (p 5) – Un composant indépendant fournissant des méthodes déjà implémentées, facilitant le développement de d'une application plus importante.

Démonstrateur (p 5) – Un démonstrateur est un logiciel simple, permettant de montrer les possibilités d'une bibliothèque

Diagramme de cas d'utilisation (p 5) – Représentation graphique permettant de décrire les intéractions entre les acteurs et le système.

Diagramme de classe (p 5) – Schéma utilisé en génie logiciel pour représenter les classes et les interfaces des systèmes ainsi que les différentes relations entre celles-ci.

Diagramme de séquence (p 5) – Représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique. Ce diagramme est inclus dans la partie dynamique d'UML.

JUnit (p 7) – Bibliothèque permettant de créer des tests unitaires simplement

UML (p 5) – (Unified Modeling Language) Langage de modélisation graphique à base de pictogramme. Il est apparu dans le monde du génie logiciel dans le cadre de la conception orientée objet. Ce langage est composé de différents diagrammes, allant du développement à la simple analyse des besoins.

Annexe B

Documentation des tests unitaires

diagramme

Class DiagrammeCasUtilisationTest

java.lang.Object
  └ **diagramme.DiagrammeCasUtilisationTest**

public class **DiagrammeCasUtilisationTest**
extends java.lang.Object

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe DiagrammeCasUtilisationTest

See Also:

DiagrammeCasUtilisation

Constructor Summary

	DiagrammeCasUtilisationTest()

Method Summary

void	setUp() Initialisation des champ avant chaque test
void	tearDown() Suppression de ces champs après chaque test
void	testEltAutoriseActeurActif() Test unitaire qui vérifie que l'élément acteur actif est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne vrai
void	testEltAutoriseActeurPassif() Test unitaire qui vérifie que l'élément acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne faux
void	testEltAutoriseCasUtilisation() Test unitaire qui vérifie que l'élément cas d'utilisation est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne vrai
void	testEltAutoriseClasse() Test unitaire qui vérifie que l'élément classe n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne faux
void	testEltAutoriseInterface() Test unitaire qui vérifie que l'élément Interface n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne faux
void	testEltAutoriseLien() Test unitaire qui vérifie que l'élément lien est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne vrai
void	testEltAutoriseTraitement() Test unitaire qui vérifie que l'élément traitement n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne faux
void	testLienAutoriseAgregation() Test unitaire qui vérifie que le lien d'agrégation entre n'importe quel élément graphique n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAssociationActeurActifActeurActif() Test unitaire qui vérifie que le lien d'association entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAssociationActeurActifActeurPassif() Test unitaire qui vérifie que le lien d'association entre un acteur actif et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAssociationActeurActifCasUtilisation() Test unitaire qui vérifie que le lien d'association entre un acteur actif et un cas d'utilisation est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne vrai
void	testLienAutoriseAssociationActeurPassifActeurActif() Test unitaire qui vérifie que le lien d'association entre un acteur passif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAssociationActeurPassifActeurPassif() Test unitaire qui vérifie que le lien d'association entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAssociationActeurPassifCasUtilisation() Test unitaire qui vérifie que le lien d'association entre un acteur passif et un cas d'utilisation n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAssociationCasUtilisationActeurActif() Test unitaire qui vérifie que le lien d'association entre un cas d'utilisation et un acteur actif est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne vrai
void	testLienAutoriseAssociationCasUtilisationActeurPassif() Test unitaire qui vérifie que le lien d'association entre un cas d'utilisation et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAssociationCasUtilisationCasUtilisation() Test unitaire qui vérifie que le lien d'association entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne vrai
void	testLienAutoriseClasseAssociation() Test unitaire qui vérifie que le lien de classe-association entre n'importe quel élément graphique n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseComposition() Test unitaire qui vérifie que le lien de composition entre n'importe quel élément graphique n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseDependanceActeurActifActeurActif() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseDependanceActeurActifActeurPassif() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur actif et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseDependanceActeurActifCasUtilisation() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur actif et un cas d'utilisation n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseDependanceActeurPassifActeurActif() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur passif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseDependanceActeurPassifCasUtilisation() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur passif et un cas d'utilisation n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseDependanceCasUtilisationActeurActif() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un cas d'utilisation et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseDependanceCasUtilisationActeurPassif() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un cas d'utilisation et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseDependanceCasUtilisationCasUtilisation() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne vrai
void	testLienAutoriseFleche() Test unitaire qui vérifie que le lien de fleche entre n'importe quel élément graphique n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseGeneralisationActeurActifActeurActif() Test unitaire qui vérifie que le lien de généralisation entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseGeneralisationActeurActifActeurPassif() Test unitaire qui vérifie que le lien de généralisation entre un acteur actif et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseGeneralisationActeurActifCasUtilisation() Test unitaire qui vérifie que le lien de généralisation entre un acteur actif et un cas d'utilisation n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne

	faux
void	testLienAutoriseGeneralisationActeurPassifActeurActif() Test unitaire qui vérifie que le lien de généralisation entre un acteur passif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseGeneralisationActeurPassifActeurPassif() Test unitaire qui vérifie que le lien de généralisation entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseGeneralisationActeurPassifCasUtilisation() Test unitaire qui vérifie que le lien de généralisation entre un acteur passif et un cas d'utilisation n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseGeneralisationCasUtilisationActeurActif() Test unitaire qui vérifie que le lien de généralisation entre un cas d'utilisation et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseGeneralisationCasUtilisationActeurPassif() Test unitaire qui vérifie que le lien de généralisation entre un cas d'utilisation et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseGeneralisationCasUtilisationCasUtilisation() Test unitaire qui vérifie que le lien de généralisation entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne vrai
void	testLienAutoriseSpecialisation() Test unitaire qui vérifie que le lien de spécialisation entre n'importe quel élément graphique n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

DiagrammeCasUtilisationTest

public **DiagrammeCasUtilisationTest**()

Method Detail

setUp

public void **setUp**()

Initialisation des champ avant chaque test

---

tearDown

public void **tearDown**()

Suppression de ces champs après chaque test

---

testLienAutoriseAssociationActeurActifCasUtilisation

public void **testLienAutoriseAssociationActeurActifCasUtilisation**()

Test unitaire qui vérifie que le lien d'association entre un acteur actif et un cas d'utilisation est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationCasUtilisationActeurActif

public void **testLienAutoriseAssociationCasUtilisationActeurActif**()

Test unitaire qui vérifie que le lien d'association entre un cas d'utilisation et un acteur actif est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationCasUtilisationCasUtilisation

public void **testLienAutoriseAssociationCasUtilisationCasUtilisation**()

Test unitaire qui vérifie que le lien d'association entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationActeurActifActeurActif

public void **testLienAutoriseAssociationActeurActifActeurActif**()

Test unitaire qui vérifie que le lien d'association entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationActeurPassifActeurPassif

public void **testLienAutoriseAssociationActeurPassifActeurPassif**()

Test unitaire qui vérifie que le lien d'association entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationActeurPassifCasUtilisation

public void **testLienAutoriseAssociationActeurPassifCasUtilisation**()

Test unitaire qui vérifie que le lien d'association entre un acteur passif et un cas d'utilisation n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationCasUtilisationActeurPassif

public void **testLienAutoriseAssociationCasUtilisationActeurPassif**()

Test unitaire qui vérifie que le lien d'association entre un cas d'utilisation et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationActeurActifActeurPassif

public void **testLienAutoriseAssociationActeurActifActeurPassif**()

Test unitaire qui vérifie que le lien d'association entre un acteur actif et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationActeurPassifActeurActif

public void **testLienAutoriseAssociationActeurPassifActeurActif**()

Test unitaire qui vérifie que le lien d'association entre un acteur passif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisationCasUtilisationCasUtilisation

public void **testLienAutoriseGeneralisationCasUtilisationCasUtilisation**()

Test unitaire qui vérifie que le lien de généralisation entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseGeneralisationActeurActifCasUtilisation

public void **testLienAutoriseGeneralisationActeurActifCasUtilisation**()

Test unitaire qui vérifie que le lien de généralisation entre un acteur actif et un cas d'utilisation n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisationCasUtilisationActeurActif

public void **testLienAutoriseGeneralisationCasUtilisationActeurActif**()

Test unitaire qui vérifie que le lien de généralisation entre un cas d'utilisation et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisationActeurPassifCasUtilisation

public void **testLienAutoriseGeneralisationActeurPassifCasUtilisation**()

Test unitaire qui vérifie que le lien de généralisation entre un acteur passif et un cas d'utilisation n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisationCasUtilisationActeurPassif

public void **testLienAutoriseGeneralisationCasUtilisationActeurPassif**()

Test unitaire qui vérifie que le lien de généralisation entre un cas d'utilisation et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisationActeurActifActeurActif

public void **testLienAutoriseGeneralisationActeurActifActeurActif**()

Test unitaire qui vérifie que le lien de généralisation entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisationActeurPassifActeurPassif

public void **testLienAutoriseGeneralisationActeurPassifActeurPassif**()

Test unitaire qui vérifie que le lien de généralisation entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisationActeurActifActeurPassif

public void **testLienAutoriseGeneralisationActeurActifActeurPassif**()

Test unitaire qui vérifie que le lien de généralisation entre un acteur actif et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisationActeurPassifActeurActif

public void **testLienAutoriseGeneralisationActeurPassifActeurActif**()

Test unitaire qui vérifie que le lien de généralisation entre un acteur passif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceCasUtilisationCasUtilisation

public void **testLienAutoriseDependanceCasUtilisationCasUtilisation**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceActeurActifActeurActif

public void **testLienAutoriseDependanceActeurActifActeurActif**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceActeurActifCasUtilisation

public void **testLienAutoriseDependanceActeurActifCasUtilisation**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur actif et un cas d'utilisation n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceCasUtilisationActeurActif

public void **testLienAutoriseDependanceCasUtilisationActeurActif**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un cas d'utilisation et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceActeurPassifCasUtilisation

public void **testLienAutoriseDependanceActeurPassifCasUtilisation**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur passif et un cas d'utilisation n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceCasUtilisationActeurPassif

public void **testLienAutoriseDependanceCasUtilisationActeurPassif**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un cas d'utilisation et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceActeurActifActeurPassif

public void **testLienAutoriseDependanceActeurActifActeurPassif**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceActeurPassifActeurActif

public void **testLienAutoriseDependanceActeurPassifActeurActif**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur passif et un acteur actif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAgregation

public void **testLienAutoriseAgregation**()

Test unitaire qui vérifie que le lien d'agrégation entre n'importe quel élément graphique n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseClasseAssociation

public void **testLienAutoriseClasseAssociation**()

Test unitaire qui vérifie que le lien de classe-association entre n'importe quel élément graphique n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseComposition

public void **testLienAutoriseComposition**()

Test unitaire qui vérifie que le lien de composition entre n'importe quel élément graphique n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseSpecialisation

public void **testLienAutoriseSpecialisation**()

Test unitaire qui vérifie que le lien de spécialisation entre n'importe quel élément graphique n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseFleche

public void **testLienAutoriseFleche**()

Test unitaire qui vérifie que le lien de fleche entre n'importe quel élément graphique n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode lienAutorise() retourne faux

---

testEltAutoriseCasUtilisation

public void **testEltAutoriseCasUtilisation**()

Test unitaire qui vérifie que l'élément cas d'utilisation est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseLien

public void **testEltAutoriseLien**()

Test unitaire qui vérifie que l'élément lien est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseActeurActif

public void **testEltAutoriseActeurActif**()

Test unitaire qui vérifie que l'élément acteur actif est autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseActeurPassif

public void **testEltAutoriseActeurPassif**()

Test unitaire qui vérifie que l'élément acteur passif n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne faux

---

testEltAutoriseTraitement

public void **testEltAutoriseTraitement**()

Test unitaire qui vérifie que l'élément traitement n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne faux

---

testEltAutoriseClasse

public void **testEltAutoriseClasse**()

Test unitaire qui vérifie que l'élément classe n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne faux

---

testEltAutoriseInterface

public void **testEltAutoriseInterface**()

Test unitaire qui vérifie que l'élément Interface n'est pas autorisé dans un diagramme de cas d'utilisation Vérifie que : - la méthode eltAutorise() retourne faux

---

---

diagramme

Class DiagrammeClasseTest

java.lang.Object
  └ **diagramme.DiagrammeClasseTest**

---

public class **DiagrammeClasseTest**
extends java.lang.Object

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe DiagrammeClasseTest

See Also:

DiagrammeClasse

Constructor Summary

DiagrammeClasseTest()

Method Summary

void	setUp() Initialisation des champ avant chaque test
void	tearDown() Suppression de ces champs après chaque test
void	testEltAutoriseActeurActif() Test unitaire qui vérifie que l'élément acteur actif n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne faux
void	testEltAutoriseActeurPassif() Test unitaire qui vérifie que l'élément acteur passif n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne faux
void	testEltAutoriseCasUtilisation() Test unitaire qui vérifie que l'élément cas d'utilisation n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne faux
void	testEltAutoriseClasse() Test unitaire qui vérifie que l'élément classe est autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne vrai
void	testEltAutoriseInterface() Test unitaire qui vérifie que l'élément interface n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne faux
void	testEltAutoriseLien() Test unitaire qui vérifie que l'élément lien est autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne vrai
void	testEltAutoriseTraitement() Test unitaire qui vérifie que l'élément traitement n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne faux
void	testLienAutoriseAgregationClasseClasse() Test unitaire qui vérifie que le lien d'agrégation entre une classe et une classe n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAgregationClasseLien() Test unitaire qui vérifie que le lien d'agrégation entre une classe et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAgregationLienClasse() Test unitaire qui vérifie que le lien d'agrégation entre un lien et une classe n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAgregationLienLien() Test unitaire qui vérifie que le lien d'agrégation entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAssociationClasseClasse() Test unitaire qui vérifie que le lien d'association entre une classe et une classe est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai
void	testLienAutoriseAssociationClasseLien() Test unitaire qui vérifie que le lien d'association entre une classe et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAssociationLienClasse() Test unitaire qui vérifie que le lien d'association entre un lien et une classe n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseAssociationLienLien() Test unitaire qui vérifie que le lien d'association entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseClaseeAssociationLienLien() Test unitaire qui vérifie que le lien de classe association entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseClasseAssociationLienClasse() Test unitaire qui vérifie que le lien de classe association entre une classe et un lien est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai
void	testLienAutoriseCompositionClasseClasse() Test unitaire qui vérifie que le lien de composition entre une classe et une classe est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai
void	testLienAutoriseCompositionClasseLien() Test unitaire qui vérifie que le lien de composition entre une classe et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseCompositionLienClasse() Test unitaire qui vérifie que le lien de composition entre un lien et une classe n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseCompositionLienLien() Test unitaire qui vérifie que le lien de composition entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	testLienAutoriseDependanceClasseClasse() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre une classe et une classe est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai
void	testLienAutoriseDependanceClasseLien() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre une classe et un lien est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai

void	**testLienAutoriseDependanceLienClasse**() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un lien et une classe est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai
void	**testLienAutoriseDependanceLienLien**() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	**testLienAutoriseFleche**() Test unitaire qui vérifie que le lien de flèche entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	**testLienAutoriseGeneralisationClasseClasse**() Test unitaire qui vérifie que le lien de généralisation entre une classe et une classe est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai
void	**testLienAutoriseGeneralisationClasseLien**() Test unitaire qui vérifie que le lien de généralisation entre une classe et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	**testLienAutoriseGeneralisationLienClasse**() Test unitaire qui vérifie que le lien de généralisation entre un lien et une classe n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	**testLienAutoriseGeneralisationLienLien**() Test unitaire qui vérifie que le lien de généralisation entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux
void	**testLienAutoriseSpecialisation**() Test unitaire qui vérifie que le lien de spécialisation entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

DiagrammeClasseTest

public **DiagrammeClasseTest**()

Method Detail

setUp

public void **setUp**()

Initialisation des champ avant chaque test

---

tearDown

public void **tearDown**()

Suppression de ces champs après chaque test

---

testLienAutoriseCompositionClasseClasse

public void **testLienAutoriseCompositionClasseClasse**()

Test unitaire qui vérifie que le lien de composition entre une classe et une classe est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseCompositionClasseLien

public void **testLienAutoriseCompositionClasseLien**()

Test unitaire qui vérifie que le lien de composition entre une classe et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseCompositionLienClasse

public void **testLienAutoriseCompositionLienClasse**()

Test unitaire qui vérifie que le lien de composition entre un lien et une classe n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseCompositionLienLien

public void **testLienAutoriseCompositionLienLien**()

Test unitaire qui vérifie que le lien de composition entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationClasseClasse

public void **testLienAutoriseAssociationClasseClasse**()

Test unitaire qui vérifie que le lien d'association entre une classe et une classe est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseClasseAssociationLienClasse

public void **testLienAutoriseClasseAssociationLienClasse**()

Test unitaire qui vérifie que le lien de classe association entre une classe et un lien est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseClaseeAssociationLienLien

public void **testLienAutoriseClaseeAssociationLienLien**()

Test unitaire qui vérifie que le lien de classe association entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationClasseLien

public void **testLienAutoriseAssociationClasseLien**()

Test unitaire qui vérifie que le lien d'association entre une classe et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationLienClasse

public void **testLienAutoriseAssociationLienClasse**()

Test unitaire qui vérifie que le lien d'association entre un lien et une classe n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationLienLien

public void **testLienAutoriseAssociationLienLien**()

Test unitaire qui vérifie que le lien d'association entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisationClasseClasse

public void **testLienAutoriseGeneralisationClasseClasse**()

Test unitaire qui vérifie que le lien de généralisation entre une classe et une classe est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseGeneralisationClasseLien

public void **testLienAutoriseGeneralisationClasseLien**()

Test unitaire qui vérifie que le lien de généralisation entre une classe et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisationLienClasse

public void **testLienAutoriseGeneralisationLienClasse**()

Test unitaire qui vérifie que le lien de généralisation entre un lien et une classe n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisationLienLien

public void **testLienAutoriseGeneralisationLienLien**()

Test unitaire qui vérifie que le lien de généralisation entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceClasseClasse

public void **testLienAutoriseDependanceClasseClasse**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre une classe et une classe est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceClasseLien

public void **testLienAutoriseDependanceClasseLien**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre une classe et un lien est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceLienClasse

public void **testLienAutoriseDependanceLienClasse**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un lien et une classe est autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceLienLien

public void **testLienAutoriseDependanceLienLien**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAgregationClasseClasse

public void **testLienAutoriseAgregationClasseClasse**()

Test unitaire qui vérifie que le lien d'agrégation entre une classe et une classe n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAgregationClasseLien

public void **testLienAutoriseAgregationClasseLien**()

Test unitaire qui vérifie que le lien d'agrégation entre une classe et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAgregationLienClasse

public void **testLienAutoriseAgregationLienClasse**()

Test unitaire qui vérifie que le lien d'agrégation entre un lien et une classe n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAgregationLienLien

public void **testLienAutoriseAgregationLienLien**()

Test unitaire qui vérifie que le lien d'agrégation entre un lien et un lien n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseSpecialisation

public void **testLienAutoriseSpecialisation**()

Test unitaire qui vérifie que le lien de spécialisation entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseFleche

public void **testLienAutoriseFleche**()

Test unitaire qui vérifie que le lien de flèche entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode lienAutorise() retourne faux

---

testEltAutoriseClasse

public void **testEltAutoriseClasse**()

Test unitaire qui vérifie que l'élément classe est autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseLien

public void **testEltAutoriseLien**()

Test unitaire qui vérifie que l'élément lien est autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseCasUtilisation

public void **testEltAutoriseCasUtilisation**()

Test unitaire qui vérifie que l'élément cas d'utilisation n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne faux

---

testEltAutoriseActeurActif

public void **testEltAutoriseActeurActif**()

Test unitaire qui vérifie que l'élément acteur actif n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne faux

---

testEltAutoriseActeurPassif

public void **testEltAutoriseActeurPassif**()

Test unitaire qui vérifie que l'élément acteur passif n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne faux

---

testEltAutoriseTraitement

public void **testEltAutoriseTraitement**()

Test unitaire qui vérifie que l'élément traitement n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne faux

---

testEltAutoriseInterface

public void **testEltAutoriseInterface**()

Test unitaire qui vérifie que l'élément interface n'est pas autorisé dans un diagramme de classe Vérifie que : - la méthode eltAutorise() retourne faux

diagramme

Class DiagrammeSequenceTest

java.lang.Object
  └ **diagramme.DiagrammeSequenceTest**

public class **DiagrammeSequenceTest**
extends java.lang.Object

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe DiagrammeSequenceTest

See Also:

DiagrammeSequence

Constructor Summary

	DiagrammeSequenceTest()

Method Summary

void	setUp() Initialisation des champ avant chaque test
void	tearDown() Suppression de ces champs après chaque test
void	testEltAutoriseActeurActif() Test unitaire qui vérifie que l'élément acteur actif est autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne vrai
void	testEltAutoriseActeurPassif() Test unitaire qui vérifie que l'élément acteur passif est autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne vrai
void	testEltAutoriseCasUtilisation() Test unitaire qui vérifie que l'élément cas d'utilisation n'est pas autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne faux
void	testEltAutoriseClasse() Test unitaire qui vérifie que l'élément classe n'est pas autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne faux
void	testEltAutoriseInterface() Test unitaire qui vérifie que l'élément interface n'est pas autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne faux
void	testEltAutoriseLien() Test unitaire qui vérifie que l'élément lien n'est pas autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne faux
void	testEltAutoriseTraitement() Test unitaire qui vérifie que l'élément traitement est autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne vrai
void	testLienAutoriseAgregation() Test unitaire qui vérifie que le lien d'agrégation entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseAssociationActeurActifActeurActif() Test unitaire qui vérifie que le lien d'association entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseAssociationActeurActifActeurPassif() Test unitaire qui vérifie que le lien d'association entre un acteur actif et un acteur passif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseAssociationActeurActifTraitement() Test unitaire qui vérifie que le lien d'association entre un acteur actif et un traitement n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseAssociationActeurPassifActeurActif() Test unitaire qui vérifie que le lien d'association entre un acteur passif et un acteur actif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseAssociationActeurPassifActeurPassif() Test unitaire qui vérifie que le lien d'association entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseAssociationActeurPassifTraitement() Test unitaire qui vérifie que le lien d'association entre un acteur passif et un traitement n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseAssociationTraitementActeurActif() Test unitaire qui vérifie que le lien d'association entre un traitement et un acteur actif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseAssociationTraitementActeurPassif() Test unitaire qui vérifie que le lien d'association entre un traitement et un acteur passif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseAssociationTraitementTraitement() Test unitaire qui vérifie que le lien d'association entre un traitement et un traitement est autorisé dans un diagramme de séquence.
void	testLienAutoriseClasseAssociation() Test unitaire qui vérifie que le lien de classe-association entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseComposition() Test unitaire qui vérifie que le lien de composition entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseDependanceActeurActifActeurActif() Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseDependanceActeurActifActeurPassif() Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur actif et un acteur passif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseDependanceActeurActifTraitement() Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur actif et un traitement n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseDependanceActeurPassifActeurActif() Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur passif et un acteur actif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseDependanceActeurPassifActeurPassif() Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseDependanceActeurPassifTraitement() Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur passif et un traitement n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseDependanceTraitementActeurActif() Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un traitement et un acteur actif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseDependanceTraitementActeurPassif() Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un traitement et un acteur passif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseDependanceTraitementTraitement() Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un traitement et un traitement est autorisé dans un diagramme de séquence.
void	testLienAutoriseFlecheActeurActifActeurActif() Test unitaire qui vérifie que le lien de flèche entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseFlecheActeurActifActeurPassif() Test unitaire qui vérifie que le lien de flèche entre un acteur actif et un acteur passif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseFlecheActeurActifTraitement() Test unitaire qui vérifie que le lien de flèche entre un acteur actif et un traitement n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseFlecheActeurPassifActeurActif() Test unitaire qui vérifie que le lien de flèche entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseFlecheActeurPassifActeurPassif() Test unitaire qui vérifie que le lien de flèche entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseFlecheActeurPassifTraitement() Test unitaire qui vérifie que le lien de flèche entre un acteur passif et un traitement n'est pas autorisé dans un diagramme de séquence.

void	testLienAutoriseFlecheTraitementActeurActif() Test unitaire qui vérifie que le lien de flèche entre un traitement et un acteur actif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseFlecheTraitementActeurPassif() Test unitaire qui vérifie que le lien de flèche entre un traitement et un acteur passif n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseFlecheTraitementTraitement() Test unitaire qui vérifie que le lien de flèche entre un traitement et un traitement est autorisé dans un diagramme de séquence.
void	testLienAutoriseGeneralisation() Test unitaire qui vérifie que le lien de généralisation entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de séquence.
void	testLienAutoriseSpecialisation() Test unitaire qui vérifie que le lien de spécialisation entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de séquence.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

DiagrammeSequenceTest

public **DiagrammeSequenceTest**()

Method Detail

setUp

public void **setUp**()

Initialisation des champ avant chaque test

---

tearDown

public void **tearDown**()

Suppression de ces champs après chaque test

---

testLienAutoriseAssociationTraitementTraitement

public void **testLienAutoriseAssociationTraitementTraitement**()

Test unitaire qui vérifie que le lien d'association entre un traitement et un traitement est autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationActeurActifTraitement

public void **testLienAutoriseAssociationActeurActifTraitement**()

Test unitaire qui vérifie que le lien d'association entre un acteur actif et un traitement n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationActeurPassifTraitement

public void **testLienAutoriseAssociationActeurPassifTraitement**()

Test unitaire qui vérifie que le lien d'association entre un acteur passif et un traitement n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationTraitementActeurActif

public void **testLienAutoriseAssociationTraitementActeurActif**()

Test unitaire qui vérifie que le lien d'association entre un traitement et un acteur actif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationTraitementActeurPassif

public void **testLienAutoriseAssociationTraitementActeurPassif**()

Test unitaire qui vérifie que le lien d'association entre un traitement et un acteur passif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationActeurActifActeurActif

public void **testLienAutoriseAssociationActeurActifActeurActif**()

Test unitaire qui vérifie que le lien d'association entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationActeurPassifActeurPassif

public void **testLienAutoriseAssociationActeurPassifActeurPassif**()

Test unitaire qui vérifie que le lien d'association entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationActeurActifActeurPassif

public void **testLienAutoriseAssociationActeurActifActeurPassif**()

Test unitaire qui vérifie que le lien d'association entre un acteur actif et un acteur passif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAssociationActeurPassifActeurActif

public void **testLienAutoriseAssociationActeurPassifActeurActif**()

Test unitaire qui vérifie que le lien d'association entre un acteur passif et un acteur actif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceTraitementTraitement

public void **testLienAutoriseDependanceTraitementTraitement**()

Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un traitement et un traitement est autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceActeurActifActeurActif

public void **testLienAutoriseDependanceActeurActifActeurActif**()

Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceActeurActifTraitement

public void **testLienAutoriseDependanceActeurActifTraitement**()

Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur actif et un traitement n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceActeurPassifTraitement

public void **testLienAutoriseDependanceActeurPassifTraitement**()

Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur passif et un traitement n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceTraitementActeurActif

public void **testLienAutoriseDependanceTraitementActeurActif**()

Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un traitement et un acteur actif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceTraitementActeurPassif

public void **testLienAutoriseDependanceTraitementActeurPassif**()

Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un traitement et un acteur passif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceActeurPassifActeurPassif

public void **testLienAutoriseDependanceActeurPassifActeurPassif**()

Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceActeurActifActeurPassif

public void **testLienAutoriseDependanceActeurActifActeurPassif**()

Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur actif et un acteur passif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseDependanceActeurPassifActeurActif

public void **testLienAutoriseDependanceActeurPassifActeurActif**()

Test unitaire qui vérifie que le lien de dependance fonctionnelle entre un acteur passif et un acteur actif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseFlecheTraitementTraitement

public void **testLienAutoriseFlecheTraitementTraitement**()

Test unitaire qui vérifie que le lien de flèche entre un traitement et un traitement est autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseFlecheActeurActifTraitement

public void **testLienAutoriseFlecheActeurActifTraitement**()

Test unitaire qui vérifie que le lien de flèche entre un acteur actif et un traitement n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseFlecheActeurPassifTraitement

public void **testLienAutoriseFlecheActeurPassifTraitement**()

Test unitaire qui vérifie que le lien de flèche entre un acteur passif et un traitement n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseFlecheTraitementActeurActif

public void **testLienAutoriseFlecheTraitementActeurActif**()

Test unitaire qui vérifie que le lien de flèche entre un traitement et un acteur actif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseFlecheTraitementActeurPassif

public void **testLienAutoriseFlecheTraitementActeurPassif**()

Test unitaire qui vérifie que le lien de flèche entre un traitement et un acteur passif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseFlecheActeurActifActeurActif

public void **testLienAutoriseFlecheActeurActifActeurActif**()

Test unitaire qui vérifie que le lien de flèche entre un acteur actif et un acteur actif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseFlecheActeurPassifActeurPassif

public void **testLienAutoriseFlecheActeurPassifActeurPassif**()

Test unitaire qui vérifie que le lien de flèche entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseFlecheActeurActifActeurPassif

public void **testLienAutoriseFlecheActeurActifActeurPassif**()

Test unitaire qui vérifie que le lien de flèche entre un acteur actif et un acteur passif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseFlecheActeurPassifActeurActif

public void **testLienAutoriseFlecheActeurPassifActeurActif**()

Test unitaire qui vérifie que le lien de flèche entre un acteur passif et un acteur passif n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseSpecialisation

public void **testLienAutoriseSpecialisation**()

Test unitaire qui vérifie que le lien de spécialisation entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseAgregation

public void **testLienAutoriseAgregation**()

Test unitaire qui vérifie que le lien d'agrégation entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseClasseAssociation

public void **testLienAutoriseClasseAssociation**()

Test unitaire qui vérifie que le lien de classe-association entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseGeneralisation

public void **testLienAutoriseGeneralisation**()

Test unitaire qui vérifie que le lien de généralisation entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testLienAutoriseComposition

public void **testLienAutoriseComposition**()

Test unitaire qui vérifie que le lien de composition entre n'importe quels éléments graphiques n'est pas autorisé dans un diagramme de séquence. Vérifie que : - la méthode lienAutorise() retourne faux

---

testEltAutoriseTraitement

public void **testEltAutoriseTraitement**()

Test unitaire qui vérifie que l'élément traitement est autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseClasse

public void **testEltAutoriseClasse**()

Test unitaire qui vérifie que l'élément classe n'est pas autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne faux

---

testEltAutoriseLien

public void **testEltAutoriseLien**()

Test unitaire qui vérifie que l'élément lien n'est pas autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne faux

---

testEltAutoriseActeurActif

public void **testEltAutoriseActeurActif**()

Test unitaire qui vérifie que l'élément acteur actif est autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseActeurPassif

public void **testEltAutoriseActeurPassif**()

Test unitaire qui vérifie que l'élément acteur passif est autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseInterface

public void **testEltAutoriseInterface**()

Test unitaire qui vérifie que l'élément interface n'est pas autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne faux

---

testEltAutoriseCasUtilisation

public void **testEltAutoriseCasUtilisation**()

Test unitaire qui vérifie que l'élément cas d'utilisation n'est pas autorisé dans un diagramme de séquence Vérifie que : - la méthode eltAutorise() retourne faux

---

---

diagramme

Class DiagrammeTest

java.lang.Object
  └ **diagramme.DiagrammeTest**

---

public class **DiagrammeTest**
extends java.lang.Object

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe DiagrammeTest

See Also:

Diagramme

Constructor Summary

DiagrammeTest()

Method Summary

void	getElementGraphiqueViaCelluleActeurActif() Test unitaire qui vérifie que qu'un élément graphique acteur actif est bien relié à sa cellule.
void	getElementGraphiqueViaCelluleActeurPassif() Test unitaire qui vérifie que qu'un élément graphique acteur passif est bien relié à sa cellule.
void	getElementGraphiqueViaCelluleCasUtilisation() Test unitaire qui vérifie que qu'un élément graphique cas d'utilisation est bien relié à sa cellule.
void	getElementGraphiqueViaCelluleClasse() Test unitaire qui vérifie que qu'un élément graphique classe est bien relié à sa cellule.
void	getElementGraphiqueViaCelluleLien() Test unitaire qui vérifie que qu'un élément graphique lien est bien relié à sa cellule.
void	getElementGraphiqueViaCelluleTraitement() Test unitaire qui vérifie que qu'un élément graphique traitement
	est bien relié à sa cellule.
void	getElementGreaphiqueViaCelluleInterface() Test unitaire qui vérifie que qu'un élément graphique interface est bien relié à sa cellule.
void	setUp() Suppression de ces champs après chaque test
void	tearDown() Suppression de ces champs après chaque test
void	testConstructeurDiagramme() Test unitaire qui vérifie que l'instanciation de la liste d'éléments graphiques.
void	testEltAutoriseActeurActif() Test unitaire qui vérifie que l'élément acteur actif est autorisé dans un diagramme.
void	testEltAutoriseActeurPassif() Test unitaire qui vérifie que l'élément acteur passif est autorisé dans un diagramme.
void	testEltAutoriseCasUtilisation() Test unitaire qui vérifie que l'élément cas d'utilisation est autorisé dans un diagramme.
void	testEltAutoriseClasse() Test unitaire qui vérifie que l'élément classe est autorisé dans un diagramme.
void	testEltAutoriseInterface() Test unitaire qui vérifie que l'élément interface est autorisé dans un diagramme.
void	testEltAutoriseLien() Test unitaire qui vérifie que l'élément lien est autorisé dans un diagramme.
void	testEltAutoriseTraitement() Test unitaire qui vérifie que l'élément traitement est autorisé dans un diagramme.
void	testGetElementsGraphiques() Test unitaire qui vérifie que l'instanciation de la liste d'éléments graphiques.
void	testLienAutoriseAgregationClasseClasse() Test unitaire qui vérifie que le lien d'agrégation entre une classe et une classe est autorisé dans un diagramme.
void	testLienAutoriseAgregationClasseLien() Test unitaire qui vérifie que le lien d'agrégation entre une classe et une classe est autorisé dans un diagramme.
void	testLienAutoriseAgregationLienClasse() Test unitaire qui vérifie que le lien d'agrégation entre un lien et une classe est autorisé dans un diagramme.
void	testLienAutoriseAgregationLienLien() Test unitaire qui vérifie que le lien d'agrégation entre un lien et un lien est autorisé dans un diagramme.
void	testLienAutoriseAssociationActeurActifActeurActif() Test unitaire qui vérifie que le lien d'association entre un acteur actif et un acteur actif est autorisé dans un diagramme.
void	testLienAutoriseAssociationActeurActifCasUtilisation() Test unitaire qui vérifie que le lien d'association entre un acteur actif et un cas d'utilisation est autorisé dans un diagramme.
void	testLienAutoriseAssociationActeurPassifActeurPassif() Test unitaire qui vérifie que le lien d'association entre un acteur passif et un acteur passif est autorisé dans un diagramme.
void	testLienAutoriseAssociationCasUtilisationActeurActif() Test unitaire qui vérifie que le lien d'association entre un cas d'utilisation et un acteur actif est autorisé dans un diagramme.
void	testLienAutoriseAssociationCasUtilisationCasUtilisation() Test unitaire qui vérifie que le lien d'association entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme.
void	testLienAutoriseAssociationClasseClasse() Test unitaire qui vérifie que le lien d'association entre une classe et une classe est autorisé dans un diagramme.
void	testLienAutoriseAssociationClasseLien() Test unitaire qui vérifie que le lien d'association entre une classe et un lien est autorisé dans un diagramme.
void	testLienAutoriseAssociationLienClasse() Test unitaire qui vérifie que le lien d'association entre un lien et une classe est autorisé dans un diagramme.
void	testLienAutoriseAssociationLienLien() Test unitaire qui vérifie que le lien d'association entre un lien et un lien est autorisé dans un diagramme.
void	testLienAutoriseAssociationTraitementTraitement() Test unitaire qui vérifie que le lien d'association entre un traitement et un traitement est autorisé dans un diagramme.

void	testLienAutoriseClasseAssociation() Test unitaire qui vérifie que le lien de classe association entre n'importe quels éléments graphiques est autorisé dans un diagramme.
void	testLienAutoriseCompositionClasseClasse() Test unitaire qui vérifie que le lien de composition entre une classe et une classe est autorisé dans un diagramme.
void	testLienAutoriseCompositionClasseLien() Test unitaire qui vérifie que le lien de composition entre une classe et un lien est autorisé dans un diagramme.
void	testLienAutoriseCompositionLienClasse() Test unitaire qui vérifie que le lien de composition entre un lien et une classe est autorisé dans un diagramme.
void	testLienAutoriseCompositionLienLien() Test unitaire qui vérifie que le lien de composition entre un lien et un lien est autorisé dans un diagramme.
void	testLienAutoriseDepencanceActeurActifActeurActif() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur actif et un acteur actif est autorisé dans un diagramme.
void	testLienAutoriseDependanceActeurActifActeurActif() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur actif et un acteur actif est autorisé dans un diagramme.
void	testLienAutoriseDependanceCasUtilisationCasUtilisation() Test unitaire qui vérifie que le lien de dépendance foncionnelle entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme.
void	testLienAutoriseDependanceClasseClasse() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre une classe et une classe est autorisé dans un diagramme.
void	testLienAutoriseDependanceClasseLien() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre une classe et un lien est autorisé dans un diagramme.
void	testLienAutoriseDependanceLienClasse() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un lien et une classe est autorisé dans un diagramme.
void	testLienAutoriseDependanceLienLien() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un lien et un lien est autorisé dans un diagramme.
void	testLienAutoriseDependanceTraitementTraitement() Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un traitement et un traitement est autorisé dans un diagramme.

void	testLienAutoriseFleche() Test unitaire qui vérifie que le lien de flèche entre n'importe quels éléments graphiques est autorisé dans un diagramme.
void	testLienAutoriseGeneralisationActeurActifActeurActif() Test unitaire qui vérifie que le lien de généralisation entre un acteur actif et un acteur actif est autorisé dans un diagramme.
void	testLienAutoriseGeneralisationCasUtilisationCasUtilisation() Test unitaire qui vérifie que le lien de généralisation entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme.
void	testLienAutoriseGeneralisationClasseClasse() Test unitaire qui vérifie que le lien de généralisation entre une classe et une classe est autorisé dans un diagramme.
void	testLienAutoriseGeneralisationClasseLien() Test unitaire qui vérifie que le lien de généralisation entre une classe et un lien est autorisé dans un diagramme.
void	testLienAutoriseGeneralisationLienClasse() Test unitaire qui vérifie que le lien de généralisation entre un lien et une classe est autorisé dans un diagramme.
void	testLienAutoriseGeneralisationLienLien() Test unitaire qui vérifie que le lien de généralisation entre un lien et un lien est autorisé dans un diagramme.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

DiagrammeTest

public **DiagrammeTest**()

Method Detail

setUp

public void **setUp**()

Suppression de ces champs après chaque test

---

tearDown

public void **tearDown**()

Suppression de ces champs après chaque test

---

testConstructeurDiagramme

public void **testConstructeurDiagramme**()

Test unitaire qui vérifie que l'instanciation de la liste d'éléments graphiques. Vérifie que : - la liste d'éléments graphiques est une instance de liste

---

testLienAutoriseAssociationActeurActifCasUtilisation

public void **testLienAutoriseAssociationActeurActifCasUtilisation**()

Test unitaire qui vérifie que le lien d'association entre un acteur actif et un cas d'utilisation est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationCasUtilisationActeurActif

public void **testLienAutoriseAssociationCasUtilisationActeurActif**()

Test unitaire qui vérifie que le lien d'association entre un cas d'utilisation et un acteur actif est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationCasUtilisationCasUtilisation

public void **testLienAutoriseAssociationCasUtilisationCasUtilisation**()

Test unitaire qui vérifie que le lien d'association entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationActeurActifActeurActif

public void **testLienAutoriseAssociationActeurActifActeurActif**()

Test unitaire qui vérifie que le lien d'association entre un acteur actif et un acteur actif est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationActeurPassifActeurPassif

public void **testLienAutoriseAssociationActeurPassifActeurPassif**()

Test unitaire qui vérifie que le lien d'association entre un acteur passif et un acteur passif est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseGeneralisationCasUtilisationCasUtilisation

public void **testLienAutoriseGeneralisationCasUtilisationCasUtilisation**()

Test unitaire qui vérifie que le lien de généralisation entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseGeneralisationActeurActifActeurActif

public void **testLienAutoriseGeneralisationActeurActifActeurActif**()

Test unitaire qui vérifie que le lien de généralisation entre un acteur actif et un acteur actif est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceCasUtilisationCasUtilisation

public void **testLienAutoriseDependanceCasUtilisationCasUtilisation**()

Test unitaire qui vérifie que le lien de dépendance foncionnelle entre un cas d'utilisation et un cas d'utilisation est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDepencanceActeurActifActeurActif

public void **testLienAutoriseDepencanceActeurActifActeurActif**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un

acteur actif et un acteur actif est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseCompositionClasseClasse

`public void testLienAutoriseCompositionClasseClasse()`

Test unitaire qui vérifie que le lien de composition entre une classe et une classe est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseCompositionClasseLien

`public void testLienAutoriseCompositionClasseLien()`

Test unitaire qui vérifie que le lien de composition entre une classe et un lien est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseCompositionLienClasse

`public void testLienAutoriseCompositionLienClasse()`

Test unitaire qui vérifie que le lien de composition entre un lien et une classe est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseCompositionLienLien

`public void testLienAutoriseCompositionLienLien()`

Test unitaire qui vérifie que le lien de composition entre un lien et un lien est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationClasseClasse

`public void testLienAutoriseAssociationClasseClasse()`

Test unitaire qui vérifie que le lien d'association entre une classe et une classe est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationClasseLien

`public void testLienAutoriseAssociationClasseLien()`

Test unitaire qui vérifie que le lien d'association entre une classe et un lien est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationLienClasse

`public void testLienAutoriseAssociationLienClasse()`

Test unitaire qui vérifie que le lien d'association entre un lien et une classe est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationLienLien

`public void testLienAutoriseAssociationLienLien()`

Test unitaire qui vérifie que le lien d'association entre un lien et un lien est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseGeneralisationClasseClasse

`public void testLienAutoriseGeneralisationClasseClasse()`

Test unitaire qui vérifie que le lien de généralisation entre une classe et une classe est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseGeneralisationClasseLien

`public void testLienAutoriseGeneralisationClasseLien()`

Test unitaire qui vérifie que le lien de généralisation entre une classe et un lien est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseGeneralisationLienClasse

`public void testLienAutoriseGeneralisationLienClasse()`

Test unitaire qui vérifie que le lien de généralisation entre un lien et une classe est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseGeneralisationLienLien

`public void testLienAutoriseGeneralisationLienLien()`

Test unitaire qui vérifie que le lien de généralisation entre un lien et un lien est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceClasseClasse

`public void testLienAutoriseDependanceClasseClasse()`

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre une classe et une classe est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceClasseLien

`public void testLienAutoriseDependanceClasseLien()`

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre une classe et un lien est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceLienClasse

`public void testLienAutoriseDependanceLienClasse()`

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un lien et une classe est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceLienLien

`public void testLienAutoriseDependanceLienLien()`

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un lien et un lien est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAgregationClasseClasse

`public void testLienAutoriseAgregationClasseClasse()`

Test unitaire qui vérifie que le lien d'agrégation entre une classe et une classe est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAgregationClasseLien

`public void testLienAutoriseAgregationClasseLien()`

Test unitaire qui vérifie que le lien d'agrégation entre une classe et une classe est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAgregationLienClasse

`public void testLienAutoriseAgregationLienClasse()`

Test unitaire qui vérifie que le lien d'agrégation entre un lien et une classe est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAgregationLienLien

`public void testLienAutoriseAgregationLienLien()`

Test unitaire qui vérifie que le lien d'agrégation entre un lien et un lien est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseAssociationTraitementTraitement

public void **testLienAutoriseAssociationTraitementTraitement**()

Test unitaire qui vérifie que le lien d'association entre un traitement et un traitement est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceTraitementTraitement

public void **testLienAutoriseDependanceTraitementTraitement**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un traitement et un traitement est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseDependanceActeurActifActeurActif

public void **testLienAutoriseDependanceActeurActifActeurActif**()

Test unitaire qui vérifie que le lien de dépendance fonctionnelle entre un acteur actif et un acteur actif est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseClasseAssociation

public void **testLienAutoriseClasseAssociation**()

Test unitaire qui vérifie que le lien de classe association entre n'importe quels éléments graphiques est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testLienAutoriseFleche

public void **testLienAutoriseFleche**()

Test unitaire qui vérifie que le lien de flèche entre n'importe quels éléments graphiques est autorisé dans un diagramme. Vérifie que : - la méthode lienAutorise() retourne vrai

---

testEltAutoriseTraitement

public void **testEltAutoriseTraitement**()

Test unitaire qui vérifie que l'élément traitement est autorisé dans un diagramme. Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseClasse

public void **testEltAutoriseClasse**()

Test unitaire qui vérifie que l'élément classe est autorisé dans un diagramme. Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseLien

public void **testEltAutoriseLien**()

Test unitaire qui vérifie que l'élément lien est autorisé dans un diagramme. Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseActeurActif

public void **testEltAutoriseActeurActif**()

Test unitaire qui vérifie que l'élément acteur actif est autorisé dans un diagramme. Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseActeurPassif

public void **testEltAutoriseActeurPassif**()

Test unitaire qui vérifie que l'élément acteur passif est autorisé dans un diagramme. Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseInterface

public void **testEltAutoriseInterface**()

Test unitaire qui vérifie que l'élément interface est autorisé dans un diagramme. Vérifie que : - la méthode eltAutorise() retourne vrai

---

testEltAutoriseCasUtilisation

public void **testEltAutoriseCasUtilisation**()

Test unitaire qui vérifie que l'élément cas d'utilisation est autorisé dans un diagramme. Vérifie que : - la méthode eltAutorise() retourne vrai

---

testGetElementsGraphiques

public void **testGetElementsGraphiques**()

Test unitaire qui vérifie que l'instanciation de la liste d'éléments graphiques. Vérifie que : - la liste d'éléments graphiques est une liste d'éléments graphiques vide

---

getElementGraphiqueViaCelluleTraitement

public void **getElementGraphiqueViaCelluleTraitement**()

Test unitaire qui vérifie que qu'un élément graphique traitement est bien relié à sa cellule. Vérifie que : - la méthode getElementGraphiqueViaCellule retourne bien l'élément graphique dont la cellule est mise en paramètre.

---

getElementGraphiqueViaCelluleClasse

public void **getElementGraphiqueViaCelluleClasse**()

Test unitaire qui vérifie que qu'un élément graphique classe est bien relié à sa cellule. Vérifie que : - la méthode getElementGraphiqueViaCellule retourne bien l'élément graphique dont la cellule est mise en paramètre.

---

getElementGraphiqueViaCelluleLien

public void **getElementGraphiqueViaCelluleLien**()

Test unitaire qui vérifie que qu'un élément graphique lien est bien relié à sa cellule. Vérifie que : - la méthode getElementGraphiqueViaCellule retourne bien l'élément graphique dont la cellule est mise en paramètre.

---

getElementGraphiqueViaCelluleActeurActif

public void **getElementGraphiqueViaCelluleActeurActif**()

Test unitaire qui vérifie que qu'un élément graphique acteur actif est bien relié à sa cellule. Vérifie que : - la méthode getElementGraphiqueViaCellule retourne bien l'élément graphique dont la cellule est mise en paramètre.

---

getElementGraphiqueViaCelluleActeurPassif

public void **getElementGraphiqueViaCelluleActeurPassif**()

Test unitaire qui vérifie que qu'un élément graphique acteur passif est bien relié à sa cellule. Vérifie que : - la méthode getElementGraphiqueViaCellule retourne bien l'élément graphique dont la cellule est mise en paramètre.

---

getElementGreaphiqueViaCelluleInterface

public void **getElementGreaphiqueViaCelluleInterface**()

Test unitaire qui vérifie que qu'un élément graphique interface est bien relié à sa cellule. Vérifie que : - la méthode getElementGraphiqueViaCellule retourne bien l'élément graphique dont la cellule est mise en paramètre.

---

getElementGraphiqueViaCelluleCasUtilisation

public void **getElementGraphiqueViaCelluleCasUtilisation**()

Test unitaire qui vérifie que qu'un élément graphique cas d'utilisation est bien relié à sa cellule. Vérifie que : - la méthode getElementGraphiqueViaCellule retourne bien l'élément graphique dont la cellule est mise en paramètre.

---

eltGraphique

Class ActeurActifTest

java.lang.Object
  └ **eltGraphique.ActeurActifTest**

public class **ActeurActifTest**
extends java.lang.Object

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe ActeurActif

See Also:

ActeurActif

Constructor Summary

	ActeurActifTest()

Method Summary

void	setUp() Initialisation du champ avant chaque test
void	tearDown() Suppression de ce champ après chaque test
void	testApplicationStyle() Test unitaire qui vérifie l'application du style sur l'acteur actif en recupérant le style de la cellule liée à l'acteur actif Vérifie que : - le style de la cellule est ACTEUR_ACTIF
void	testCelluleParent() Test unitaire qui teste l'acteur actif n'a pas d'élément parent Vérifie que : - la cellule parente est la même que la cellule actuelle
void	testCelluleTarget() Test unitaire qui teste l'acteur actif n'a pas d'élément cible Vérifie que : - la cellule cible est la même que la cellule actuelle
void	testCreerCelluleNotNull() Test unitaire qui teste la création de la cellule en vérifiant que l'attribut n'est pas null Vérifie que : - L'attribut cellle du champ n'est pas null
void	testCreerStyle() Test unitaire qui teste la création du style de l'acteur actif en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle d'un acteur actif - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte
void	testInstance() Test unitaire que teste l'instanciation du champ Vérifie que : - le champ est une instance de la classe ActeurActif
void	testSetCellule() Test unitaire qui teste la modification de la cellule lié à l'acteur actif Vérifie que : - la cellule actuelle a bien été remplacée - la cellule cible est la même que la cellule courante - la cellule parent est la mee que la cellule courante
void	testSetTexte() Test unitaire qui teste la modification du texte lié à l'acteur actif Vérifie que : - le texte est modifié correctement
void	testSupprimer() Test unitaire qui teste al suppression de la cellule liée à l'acteur Vérifie que : - la cellule courante est null - la cellule cible est null - la cellule parent est null

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

ActeurActifTest

public **ActeurActifTest**()

Method Detail

setUp

public void **setUp**()

Initialisation du champ avant chaque test

tearDown

public void **tearDown**()

Suppression de ce champ après chaque test

testCreerStyle

public void **testCreerStyle**()

Test unitaire qui teste la création du style de l'acteur actif en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle d'un acteur actif - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte

testApplicationStyle

public void **testApplicationStyle**()

Test unitaire qui vérifie l'application du style sur l'acteur actif en recupérant le style de la cellule liée à l'acteur actif Vérifie que : - le style de la cellule est ACTEUR_ACTIF

testCreerCelluleNotNull

public void **testCreerCelluleNotNull**()

Test unitaire qui teste la création de la cellule en vérifiant que l'attribut n'est pas null Vérifie que : - L'attribut cellle du champ n'est pas null

testInstance

public void **testInstance**()

Test unitaire que teste l'instanciation du champ Vérifie que : - le champ est une instance de la classe ActeurActif

testCelluleParent

public void **testCelluleParent**()

Test unitaire qui teste l'acteur actif n'a pas d'élément parent Vérifie que : - la cellule parente est la même que la cellule actuelle

testCelluleTarget

public void **testCelluleTarget**()

Test unitaire qui teste l'acteur actif n'a pas d'élément cible Vérifie que : - la cellule cible est la même que la cellule actuelle

testSetTexte

public void **testSetTexte**()

Test unitaire qui teste la modification du texte lié à l'acteur actif Vérifie que : - le texte est modifié correctement

testSetCellule

public void **testSetCellule**()

Test unitaire qui teste la modification de la cellule lié à l'acteur actif Vérifie que : - la cellule actuelle a bien été remplacée - la cellule cible est la même que la cellule courante - la cellule parent est la mee que la cellule courante

testSupprimer

public void **testSupprimer**()

Test unitaire qui teste al suppression de la cellule liée à l'acteur Vérifie que : - la cellule courante est null - la cellule cible est null - la cellule parent est null

eltGraphique

Class ActeurPassifTest

java.lang.Object
  └ **eltGraphique.ActeurPassifTest**

public class **ActeurPassifTest**
extends java.lang.Object

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe ActeurPassif

See Also:

ActeurPassif

Constructor Summary

[ActeurPassifTest](#)()	

Method Summary

void	[setUp](#)() Initialisation du champ avant chaque test
void	[tearDown](#)() Suppression de ce champ après chaque test
void	[testCellule](#)() Test unitaire qui teste que la methode getCellule retourne bien la bonne cellule Vérifie que : - la cellule renvoyée par getCellule est correcte
void	[testCelluleParent](#)() Test unitaire qui teste l'acteur passif n'a pas d'élément parent Vérifie que : - la cellule parente est la même que la cellule actuelle
void	[testCelluleTarget](#)() Test unitaire qui teste que l'acteur passif n'a pas d'élément cible Vérifie que : - la cellule cible est la même que la cellule actuelle
void	[testCreationStyle](#)() Test unitaire qui teste la création du style de l'acteur passif en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle d'un acteur passif - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte
void	[testInstance](#)() Test unitaire que teste l'instanciation du champ Vérifie que : - le champ est une instance de la classe ActeurPassif
void	[testSetCelluleParent](#)() Test unitaire qui teste la modification de la cellule lié à l'acteur passif Vérifie que : - la cellule actuelle a bien été remplacée - la cellule parent est la même que la cellule courante
void	[testSetCelluleTarget](#)() Test unitaire qui teste la modification de la cellule lié à l'acteur passif Vérifie que : - la cellule cible est la même que la cellule courante
void	[testSetTexte](#)() Test unitaire qui teste la modification du texte lié à l'acteur passif Vérifie que : - le texte est modifié correctement
void	[testStyle](#)() Test unitaire qui vérifie l'application du style sur l'acteur passif en recupérant le style de la cellule liée à l'acteur passif Vérifie que : - le style de la cellule est ACTEUR_PASSIF
void	[testSupprimer](#)() Test unitaire qui teste al suppression de la cellule liée à l'acteur Vérifie que : - la cellule courante est null - la cellule cible est null - la cellule parent est null

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

ActeurPassifTest

public **ActeurPassifTest**()

Method Detail

setUp

public void **setUp**()

Initialisation du champ avant chaque test

tearDown

public void **tearDown**()

Suppression de ce champ après chaque test

testCelluleParent

public void **testCelluleParent**()

Test unitaire qui teste l'acteur passif n'a pas d'élément parent Vérifie que : - la cellule parente est la même que la cellule actuelle

testCelluleTarget

public void **testCelluleTarget**()

Test unitaire qui teste que l'acteur passif n'a pas d'élément cible Vérifie que : - la cellule cible est la même que la cellule actuelle

testCellule

public void **testCellule**()

Test unitaire qui teste que la methode getCellule retourne bien la bonne cellule Vérifie que : - la cellule renvoyée par getCellule est correcte

testInstance

public void **testInstance**()

Test unitaire que teste l'instanciation du champ Vérifie que : - le champ est une instance de la classe ActeurPassif

testCreationStyle

public void **testCreationStyle**()

Test unitaire qui teste la création du style de l'acteur passif en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle d'un acteur passif - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte

testStyle

public void **testStyle**()

Test unitaire qui vérifie l'application du style sur l'acteur passif en recupérant le style de la cellule liée à l'acteur passif Vérifie que : - le style de la cellule est ACTEUR_PASSIF

testSetTexte

public void **testSetTexte**()

Test unitaire qui teste la modification du texte lié à l'acteur passif Vérifie que : - le texte est modifié correctement

testSetCelluleParent

public void **testSetCelluleParent**()

Test unitaire qui teste la modification de la cellule lié à l'acteur passif Vérifie que : - la cellule actuelle a bien été remplacée - la cellule parent est la même que la cellule courante

testSetCelluleTarget

public void **testSetCelluleTarget**()

Test unitaire qui teste la modification de la cellule lié à l'acteur passif Vérifie que : - la cellule cible est la même que la cellule courante

testSupprimer

public void **testSupprimer**()

Test unitaire qui teste al suppression de la cellule liée à l'acteur Vérifie que : - la cellule courante est null - la cellule cible est null - la cellule parent est null

---

eltGraphique

Class AttributTest

```
java.lang.Object
  └ eltGraphique.AttributTest
```

```
public class AttributTest
extends java.lang.Object
```

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe Attribut

See Also:

`Attribut`

Constructor Summary

`AttributTest()`

Method Summary

`void`	`setUp()` Initialisation du champ avant chaque test
`void`	`tearDown()` Suppression de ce champ après chaque test
`void`	`testDeClasse()` Test unitaire qui teste la modification du fait que l'attribut est de classe Vérifie que : - la modification de l'attribut deClasse est effective
`void`	`testToString()` Test unitaire qui teste le retour de la méthode toString() Vérifie que : - la génération de la chaîne est correcte
`void`	`testVisibilite()` Test unitaire qui teste la modification de l'attribut visibilite Vérifie que : - la modification de l'attribut visibilite est effective

Methods inherited from class java.lang.Object

`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

AttributTest

```
public AttributTest()
```

Method Detail

setUp

```
public void setUp()
```

Initialisation du champ avant chaque test

---

tearDown

```
public void tearDown()
```

Suppression de ce champ après chaque test

---

testVisibilite

```
public void testVisibilite()
```

Test unitaire qui teste la modification de l'attribut visibilite Vérifie que : - la modification de l'attribut visibilite est effective

---

testDeClasse

```
public void testDeClasse()
```

Test unitaire qui teste la modification du fait que l'attribut est de classe Vérifie que : - la modification de l'attribut deClasse est effective

---

testToString

```
public void testToString()
```

Test unitaire qui teste le retour de la méthode toString() Vérifie que : - la génération de la chaîne est correcte

eltGraphique

Class CasUtilisationTest

java.lang.Object
  └ **eltGraphique.CasUtilisationTest**

public class **CasUtilisationTest**
extends java.lang.Object

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe CasUtilisation

See Also:

CasUtilisation

Constructor Summary

CasUtilisationTest()

Method Summary

void	**setUp**() Initialisation du champ avant chaque test
void	**tearDown**() Suppression de ce champ après chaque test
void	**testCelluleParent**() Test unitaire qui teste que le cas d'utilisation n'a pas d'élément parent Vérifie que : - la cellule parente est la même que la cellule actuelle
void	**testCelluleTarget**() Test unitaire qui teste le cas d'utilisation n'a pas d'élément cible Vérifie que : - la cellule cible est la même que la cellule actuelle
void	**testChampsStyle**() Test unitaire qui teste la création du style du cas d'utilisation en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle du cas d'utilisation - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte
void	**testInstance**() Test unitaire que teste l'instanciation du champ Vérifie que : - le champ est une instance de la classe CasUtilisation
void	**testSetCellule**() Test unitaire qui teste que la modification d'une cellule est effective Vérifie que : - la cellule actuelle est modifiée correctement - la cellule parente est modifiée correctement - la cellule cible est modifiée correctement
void	**testSetTexte**() Test unitaire qui teste que la modification du texte est effective Vérifie que : - le nouveau texte est attribué correctement
void	**testStyle**() Test unitaire qui vérifie l'application du style sur le cas d'utilisation en recupérant le style de la cellule liée à le cas d'utilisation Vérifie que : - le style de la cellule est USECASE
void	**testSupprimer**() Test unitaire qui vérifie que la méthode supprimer fonctionne correctement Vérifie que : - la cellule actuelle est à null - la cellule parente est à null - la cellule cible est à null

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

CasUtilisationTest

public **CasUtilisationTest**()

Method Detail

setUp

public void **setUp**()

Initialisation du champ avant chaque test

tearDown

public void **tearDown**()

Suppression de ce champ après chaque test

testCelluleParent

public void **testCelluleParent**()

Test unitaire qui teste que le cas d'utilisation n'a pas d'élément parent Vérifie que : - la cellule parente est la même que la cellule actuelle

testCelluleTarget

public void **testCelluleTarget**()

Test unitaire qui teste le cas d'utilisation n'a pas d'élément cible Vérifie que : - la cellule cible est la même que la cellule actuelle

testInstance

public void **testInstance**()

Test unitaire que teste l'instanciation du champ Vérifie que : - le champ est une instance de la classe CasUtilisation

testStyle

public void **testStyle**()

Test unitaire qui vérifie l'application du style sur le cas d'utilisation en recupérant le style de la cellule liée à le cas d'utilisation Vérifie que : - le style de la cellule est USECASE

testChampsStyle

public void **testChampsStyle**()

Test unitaire qui teste la création du style du cas d'utilisation en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle du cas d'utilisation - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte

testSetTexte

public void **testSetTexte**()

Test unitaire qui teste que la modification du texte est effective Vérifie que : - le nouveau texte est attribué correctement

testSetCellule

public void **testSetCellule**()

Test unitaire qui teste que la modification d'une cellule est effective Vérifie que : - la cellule actuelle est modifiée correctement - la cellule parente est modifiée correctement - la cellule cible est modifiée correctement

testSupprimer

public void **testSupprimer**()

Test unitaire qui vérifie que la méthode supprimer fonctionne correctement Vérifie que : - la cellule actuelle est à null - la cellule parente est à null - la cellule cible est à null

eltGraphique

Class ClasseTest

java.lang.Object
  └ **eltGraphique.ClasseTest**

public class **ClasseTest**
extends java.lang.Object

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe CasUtilisation

See Also:

CasUtilisation

Constructor Summary

	ClasseTest()

Method Summary

void	**setUp**() Initialisation du champ avant chaque test
void	**tearDown**() Suppression de ce champ après chaque test
void	**testAjouterAttribut**() Test unitaire qui vérifie lors de l'ajout d'un attribut à la classe que ledit est bien référencée dans la liste des attributs de la classe Vérifie que : - le nouvel attribut est présent dans la liste des attributs de la classe
void	**testAjouterAttributTailleListe**() Test unitaire qui teste la modification de la taille de la liste d'attributs lors de l'ajout d'un attribut à la classe Vérifie que : - la nouvelle taille est égale à l'ancienne + 1
void	**testAjouterMethode**() Test unitaire qui vérifie lors de l'ajout d'une méthode à la classe que ladite est bien référencée dans la liste des méthodes de la classe Vérifie que : - la nouvelle méthode est présente dans la liste des méthodes de la classe
void	**testAjouterMethodeTailleListe**() Test unitaire qui teste la modification de la taille de la liste de méthode lors de l'ajout d'une méthode à la classe Vérifie que : - la nouvelle taille est égale à l'ancienne + 1
void	**testCelluleParentTarget**() Test unitaire qui teste que la classe n'a pas d'élément parent et cible Vérifie que : - la cellule parente est la même que la cellule actuelle - la cellule cible est la même que la cellule actuelle
void	**testCreerInstance**() Test unitaire qui teste si l'instanciation du champ est correcte Vérifie que : - le champ est une instance de Classe
void	**testCreerStyle**() Test unitaire qui teste l'application du style à la cellule Vérifie que : - Le style appliqué est bien "CLASSE"
void	**testCreerStyleClasse**() Test unitaire qui teste la création du style de la classe en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle d'une classe - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte
void	**testCreerStyleContenuclasse**() Test unitaire qui teste la création du style de l'acteur passif en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle du contenu d'une classe - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte - le redimensionnement du contenu soit impossible - le déplacement du contenu soit impossible - la suppression du contenu soit impossible - l'alignement du texte est à gauche
void	**testEstAbstraite**() Test unitaire qui vérifie que l'on peut mettre une classe abstraite Vérifie que : - la modification de l'attribut abstraite de la classe est effective
void	**testEstConstante**() Test unitaire qui vérifie que l'on peut mettre une classe constante Vérifie que : - la modification de l'attribut constante de la classe est effective
void	**testGetAttributsInstance**() Test unitaire qui teste l'instanciation de la liste de méthodes de la classe Vérifie que : - la liste des méthode est une instance de Liste
void	**testGetMethodesInstance**() Test unitaire qui teste l'instanciation de la liste de attributs de la classe Vérifie que : - la liste des attributs est une instance de Liste
void	**testSetAttributs**() Test unitaire qui vérifie la modification de la liste d'attributs de la classe via la méthode setAttributs Vérifie que : - la nouvelle liste d'attributs est assignée correctement
void	**testSetCelluleParent**() Test unitaire qui teste la modification de la cellule lié à la classe Vérifie que : - la cellule actuelle a bien été remplacée - la cellule parent est la même que la cellule courante
void	**testSetCelluleTarget**() Test unitaire qui teste la modification de la cellule lié à la classe Vérifie que : - la cellule cible est la même que la cellule courante
void	**testSetMethodes**() Test unitaire qui vérifie la modification de la liste de méthodes de la classe via la méthode setMéthodes Vérifie que : - la nouvelle liste de méthodes est assignée correctement
void	**testSetTexte**() Test unitaire qui teste la modification du texte lié à la classe Vérifie que : - le texte est modifié correctement
void	**testSupprimer**() Test unitaire qui vérifie la suppression de l'élément Vérifie que : - la cellule actuelle est à null - la cellule parente est à null - la cellule cible est à null

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

ClasseTest

public **ClasseTest**()

Method Detail

setUp

public void **setUp**()

Initialisation du champ avant chaque test

tearDown

public void **tearDown**()

Suppression de ce champ après chaque test

testCelluleParentTarget

public void **testCelluleParentTarget**()

Test unitaire qui teste que la classe n'a pas d'élément parent et cible Vérifie que : - la cellule parente est la même que la cellule actuelle - la cellule cible est la même que la cellule actuelle

testCreerInstance

public void **testCreerInstance**()

Test unitaire qui teste si l'instanciation du champ est correcte Vérifie que : - le champ est une instance de Classe

testCreerStyle

public void **testCreerStyle**()

Test unitaire qui teste l'application du style à la cellule Vérifie que : - Le style appliqué est bien "CLASSE"

testSetTexte

public void **testSetTexte**()

Test unitaire qui teste la modification du texte lié à la classe Vérifie que : - le texte est modifié correctement

testSetCelluleParent

public void **testSetCelluleParent**()

Test unitaire qui teste la modification de la cellule lié à la classe Vérifie que : - la cellule actuelle a bien été remplacée - la cellule parent est la même que la cellule courante

---

testSetCelluleTarget

public void **testSetCelluleTarget**()

Test unitaire qui teste la modification de la cellule lié à la classe Vérifie que : - la cellule cible est la même que la cellule courante

---

testSupprimer

public void **testSupprimer**()

Test unitaire qui vérifie la suppression de l'élément Vérifie que : - la cellule actuelle est à null - la cellule parente est à null - la cellule cible est à null

---

testAjouterMethodeTailleListe

public void **testAjouterMethodeTailleListe**()

Test unitaire qui teste la modification de la taille de la liste de méthode lors de l'ajout d'une méthode à la classe Vérifie que : - la nouvelle taille est égale à l'ancienne + 1

---

testAjouterMethode

public void **testAjouterMethode**()

Test unitaire qui vérifie lors de l'ajout d'une méthode à la classe que ladite est bien référencée dans la liste des méthodes de la classe Vérifie que : - la nouvelle méthode est présente dans la liste des méthodes de la classe

---

testAjouterAttribut

public void **testAjouterAttribut**()

Test unitaire qui vérifie lors de l'ajout d'un attribut à la classe que ledit est bien référencée dans la liste des attributs de la classe Vérifie que : - le nouvel attribut est présent dans la liste des attributs de la classe

---

testAjouterAttributTailleListe

public void **testAjouterAttributTailleListe**()

Test unitaire qui teste la modification de la taille de la liste d'attributs lors de l'ajout d'un attribut à la classe Vérifie que : - la nouvelle taille est égale à l'ancienne + 1

---

testEstConstante

public void **testEstConstante**()

Test unitaire qui vérifie que l'on peut mettre une classe constante Vérifie que : - la modification de l'attribut constante de la classe est effective

---

testEstAbstraite

public void **testEstAbstraite**()

Test unitaire qui vérifie que l'on peut mettre une classe abstraite Vérifie que : - la modification de l'attribut abstraite de la classe est effective

---

testGetAttributsInstance

public void **testGetAttributsInstance**()

Test unitaire qui teste l'instanciation de la liste de méthodes de la classe Vérifie que : - la liste des méthode est une instance de Liste

See Also:
Liste

---

testGetMethodesInstance

public void **testGetMethodesInstance**()

Test unitaire qui teste l'instanciation de la liste de attributs de la classe

Vérifie que : - la liste des attributs est une instance de Liste

See Also:
Liste

---

testSetAttributs

public void **testSetAttributs**()

Test unitaire qui vérifie la modification de la liste d'attributs de la classe via la méthode setAttributs Vérifie que : - la nouvelle liste d'attributs est assignée correctement

---

testSetMethodes

public void **testSetMethodes**()

Test unitaire qui vérifie la modification de la liste de méthodes de la classe via la méthode setMéthodes Vérifie que : - la nouvelle liste de méthodes est assignée correctement

---

testCreerStyleClasse

public void **testCreerStyleClasse**()

Test unitaire qui teste la création du style de la classe en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle d'une classe - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte

---

testCreerStyleContenuclasse

public void **testCreerStyleContenuclasse**()

Test unitaire qui teste la création du style de l'acteur passif en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle du contenu d'une classe - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte - le redimensionnement du contenu soit impossible - le déplacement du contenu soit impossible - la suppression du contenu soit impossible - l'alignement du texte est à gauche

---

eltGraphique

Class InterfaceTest

java.lang.Object
  └ **eltGraphique.InterfaceTest**

public class **InterfaceTest**
extends java.lang.Object

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe Interface

See Also:
Interface

Constructor Summary

InterfaceTest()

Method Summary

void	setUp() Initialisation du champ avant chaque test
void	tearDown() Suppression de ce champ après chaque test
void	testFail() Les tests unitaires ne sont pas encore implémentés pour cette classe

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

InterfaceTest

public **InterfaceTest**()

Method Detail

setUp

public void **setUp**()

Initialisation du champ avant chaque test

tearDown

public void **tearDown**()

Suppression de ce champ après chaque test

testFail

public void **testFail**()

Les tests unitaires ne sont pas encore implémentés pour cette classe

eltGraphique

Class LienTest

java.lang.Object
  └ **eltGraphique.LienTest**

public class **LienTest**
extends java.lang.Object

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe Lien

See Also:
Lien

Constructor Summary

LienTest()

Method Summary

void	setUp() Initialisation du champ avant chaque test
void	tearDown() Suppression de ce champ après chaque test
void	testCreerStyleAgregation() Test unitaire qui vérifie la création du style des liens d'agrégation Vérifie que : - le style est créé - la forme est celle d'un lien d'agrégation - l'opacite est correcte - le déplacement du lien est impossible - la couleur du lien est correcte
void	testCreerStyleAssociation() Test unitaire qui vérifie la création du style des liens d'association Vérifie que : - le style est créé - la forme est celle d'un lien d'association - l'opacite est correcte - le déplacement du lien est impossible - la couleur du lien est correcte
void	testCreerStyleComposition() Test unitaire qui vérifie la création du style des liens de composition Vérifie que : - le style est créé - la forme est celle d'un lien de composition - l'opacite est correcte - le déplacement du lien est impossible - la couleur du lien est correcte
void	testCreerStyleFleche() Test unitaire qui vérifie la création du style des flèches Vérifie que : - le style est créé - la forme est celle d'une flèche - l'opacite est correcte - le déplacement du lien est impossible - la couleur du lien est correcte
void	testCreerStyleSpecialisation() Test unitaire qui vérifie la création du style des liens de spécialisation Vérifie que : - le style est créé - la forme est celle d'un lien de spécialisation - l'opacite est correcte - le déplacement du lien est impossible - la couleur du lien est correcte

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

LienTest

public **LienTest**()

Method Detail

setUp

public void **setUp**()

Initialisation du champ avant chaque test

tearDown

public void **tearDown**()

Suppression de ce champ après chaque test

testCreerStyleComposition

public void **testCreerStyleComposition**()

Test unitaire qui vérifie la création du style des liens de composition Vérifie que : - le style est créé - la forme est celle d'un lien de composition - l'opacite est correcte - le déplacement du lien est impossible - la couleur du lien est correcte

---

testCreerStyleAgregation

`public void testCreerStyleAgregation()`

Test unitaire qui vérifie la création du style des liens d'agrégation Vérifie que : - le style est créé - la forme est celle d'un lien d'agrégation - l'opacite est correcte - le déplacement du lien est impossible - la couleur du lien est correcte

---

testCreerStyleAssociation

`public void testCreerStyleAssociation()`

Test unitaire qui vérifie la création du style des liens d'association Vérifie que : - le style est créé - la forme est celle d'un lien d'association - l'opacite est correcte - le déplacement du lien est impossible - la couleur du lien est correcte

---

testCreerStyleSpecialisation

`public void testCreerStyleSpecialisation()`

Test unitaire qui vérifie la création du style des liens de spécialisation Vérifie que : - le style est créé - la forme est celle d'un lien de spécialisation - l'opacite est correcte - le déplacement du lien est impossible - la couleur du lien est correcte

---

testCreerStyleFleche

`public void testCreerStyleFleche()`

Test unitaire qui vérifie la création du style des flèches Vérifie que : - le style est créé - la forme est celle d'une flèche - l'opacite est correcte - le déplacement du lien est impossible - la couleur du lien est correcte

---

---

eltGraphique

Class MethodeTest

```
java.lang.Object
  └ eltGraphique.MethodeTest
```

`public class MethodeTest`
`extends java.lang.Object`

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe Methode

See Also:

Methode

Constructor Summary

	MethodeTest()

Method Summary

void	setUp() Initialisation du champ avant chaque test
void	tearDown() Suppression de ce champ après chaque test
void	testAbstraite() Test unitaire qui vérifie la modification de l'état abstrait d'une méthode Vérifie que : - la classe est passée abstraite
void	testAjouterParametreAvecIndex() Test unitaire qui vérifie l'ajout d'un paramètre à la liste des paramètres de la méthode avec passage de l'index Vérifie que : - le nouveau paramètre est dans la liste des paramètres de la méthode
void	testAjouterParametreAvecIndexTailleListe() Test unitaire qui teste que la taille est incrémentée de 1 après ajout d'un paramètre avec index Vérifie que : - la nouvelle taille est égale à l'ancienne + 1
void	testAjouterParametreSansIndex() Test unitaire qui vérifie l'ajout d'un paramètre à la liste des paramètres de la méthode sans passage de l'index Vérifie que : - le nouveau paramètre est dans la liste des paramètres de la méthode
void	testAjouterParametreSansIndexTailleListe() Test unitaire qui teste que la taille est incrémentée de 1 après ajout d'un paramètre sans index Vérifie que : - la nouvelle taille est égale à l'ancienne + 1
void	testConstante() Test unitaire qui vérifie la modification de l'état constant d'une méthode Vérifie que : - la classe est passée constante
void	testDeClasse() Test unitaire qui vérifie la modification de l'état de classe d'une méthode Vérifie que : - la classe est passée de classe
void	testInstanceParametre() Test unitaire qui vérifie l'instanciation de la liste des paramètres d'une méthode Vérifie que : - la liste des paramètres est une instance de Liste
void	testNom() Test unitaire qui teste la modification du nom de la méthode Vérifie que : - le texte est assigné correctement
void	testParametres() Test unitaire qui vérifie la modification de la liste des paramètres de la méthode Vérifie que : - la liste des méthodes est assignée correctement
void	testParametresTailleListe() Test unitaire qui vérifie la modification de la taille de la liste des paramètres de la classe lors de l'ajout d'un paramètre à la méthode Vérifie que : - la nouvelle taille est égale à l'ancienne + 1
void	testToString() Test unitaire qui teste le retour de la méthode toString Vérifie que : - la chaîne retournée est correcte
void	testTypeDeRetour() Test unitaire qui vérifie la modification du type de retour de la méthode Vérifie que : - le nouveau type de retour est assigné correctement
void	testVisibilite() Test unitaire qui vérifie la modification de la visibilite de la méthode Vérifie que : - la nouvelle visibilite est assignée correctement

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

MethodeTest

`public MethodeTest()`

Method Detail

setUp

`public void setUp()`

Initialisation du champ avant chaque test

---

tearDown

`public void tearDown()`

Suppression de ce champ après chaque test

---

testAjouterParametreSansIndex

`public void testAjouterParametreSansIndex()`

Test unitaire qui vérifie l'ajout d'un paramètre à la liste des paramètres de la méthode sans passage de l'index Vérifie que : - le nouveau paramètre est dans la liste des paramètres de la méthode

---

testAjouterParametreSansIndexTailleListe

`public void testAjouterParametreSansIndexTailleListe()`

Test unitaire qui teste que la taille est incrémentée de 1 après ajout d'un paramètre sans index Vérifie que : - la nouvelle taille est égale à l'ancienne + 1

---

testAjouterParametreAvecIndex

public void **testAjouterParametreAvecIndex**()

Test unitaire qui vérifie l'ajout d'un paramètre à la liste des paramètres de la méthode avec passage de l'index Vérifie que : - le nouveau paramètre est dans la liste des paramètres de la méthode

---

testAjouterParametreAvecIndexTailleListe

public void **testAjouterParametreAvecIndexTailleListe**()

Test unitaire qui teste que la taille est incrémentée de 1 après ajout d'un paramètre avec index Vérifie que : - la nouvelle taille est égale à l'ancienne + 1

---

testAbstraite

public void **testAbstraite**()

Test unitaire qui vérifie la modification de l'état abstrait d'une méthode Vérifie que : - la classe est passée abstraite

---

testConstante

public void **testConstante**()

Test unitaire qui vérifie la modification de l'état constant d'une méthode Vérifie que : - la classe est passée constante

---

testDeClasse

public void **testDeClasse**()

Test unitaire qui vérifie la modification de l'état de classe d'une méthode Vérifie que : - la classe est passée de classe

---

testNom

public void **testNom**()

Test unitaire qui teste la modification du nom de la méthode Vérifie que : - le texte est assigné correctement

---

testInstanceParametre

public void **testInstanceParametre**()

Test unitaire qui vérifie l'instanciation de la liste des paramètres d'une méthode Vérifie que : - la liste des paramètres est une instance de Liste

---

testParametres

public void **testParametres**()

Test unitaire qui vérifie la modification de la liste des paramètres de la méthode Vérifie que : - la liste des méthodes est assignée correctement

---

testParametresTailleListe

public void **testParametresTailleListe**()

Test unitaire qui vérifie la modification de la taille de la liste des paramètres de la classe lors de l'ajout d'un paramètre à la méthode Vérifie que : - la nouvelle taille est égale à l'ancienne + 1

---

testVisibilite

public void **testVisibilite**()

Test unitaire qui vérifie la modification de la visibilite de la méthode Vérifie que : - la nouvelle visibilite est assignée correctement

---

testTypeDeRetour

public void **testTypeDeRetour**()

Test unitaire qui vérifie la modification du type de retour de la méthode Vérifie que : - le nouveau type de retour est assigné correctement

---

testToString

public void **testToString**()

Test unitaire qui teste le retour de la méthode toString Vérifie que : - la chaîne retournée est correcte

---

---

eltGraphique

Class TraitementTest

java.lang.Object
  └ **eltGraphique.TraitementTest**

---

public class **TraitementTest**
extends java.lang.Object

Cas de test JUnit regroupant les tests unitaires de chaque méthode de la classe Traitement

See Also:

Traitement

Constructor Summary

TraitementTest()

Method Summary

void	setUp() Initialisation du champ avant chaque test
void	tearDown() Suppression de ce champ après chaque test
void	testCreerStyle() Test unitaire qui teste la création du style du traitement en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle d'un traitement - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte
void	testCreerStyleFleche() Test unitaire qui teste la création du style de la flèche de début de séquence en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle d'un traitement - le déplacement de la flèche soit impossible - le redimensionnement de la flèche soit impossible - la suppression de la flèche soit impossible - l'opacite est correcte - la direction est correcte - la'aligement du texte est à gauche - la couleur du texte est correcte - la couleur de fond est correcte - la

	couleur de la bordure est correcte
void	**testDebutSequence**() Test unitaire qui teste la modification du champ de début de séquence du traitement Vérifie que : - l'état initial du traitement est faux (pas début de séquence) - la modification de l'état à vrai est effective
void	**testEvenementDeclencheur**() Test unitaire qui test la modification de l'élément déclencheur du traitement Vérifie que : - le nouveau traitement est assigné correctement

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

TraitementTest

public **TraitementTest**()

Method Detail

setUp

public void **setUp**()

Initialisation du champ avant chaque test

---

tearDown

public void **tearDown**()

Suppression de ce champ après chaque test

---

testCreerStyle

public void **testCreerStyle**()

Test unitaire qui teste la création du style du traitement en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle d'un traitement - l'inclusion d'un autre élément soit impossible - l'opacite est correcte - la couleur du texte est correcte - la couleur de la bordure est correcte

---

testCreerStyleFleche

public void **testCreerStyleFleche**()

Test unitaire qui teste la création du style de la flèche de début de séquence en vérifiant chaque champ du nouveau style Vérifie que : - le style est créé - la forme est celle d'un traitement - le déplacement de la flèche soit impossible - le redimensionnement de la flèche soit impossible - la suppression de la flèche soit impossible - l'opacite est correcte - la direction est correcte - la'aligement du texte est à gauche - la couleur du texte est correcte - la couleur de fond est correcte - la couleur de la bordure est correcte

---

testEvenementDeclencheur

public void **testEvenementDeclencheur**()

Test unitaire qui test la modification de l'élément déclencheur du traitement Vérifie que : - le nouveau traitement est assigné correctement

---

testDebutSequence

public void **testDebutSequence**()

Test unitaire qui teste la modification du champ de début de séquence du traitement Vérifie que : - l'état initial du traitement est faux (pas début de séquence) - la modification de l'état à vrai est effective

---