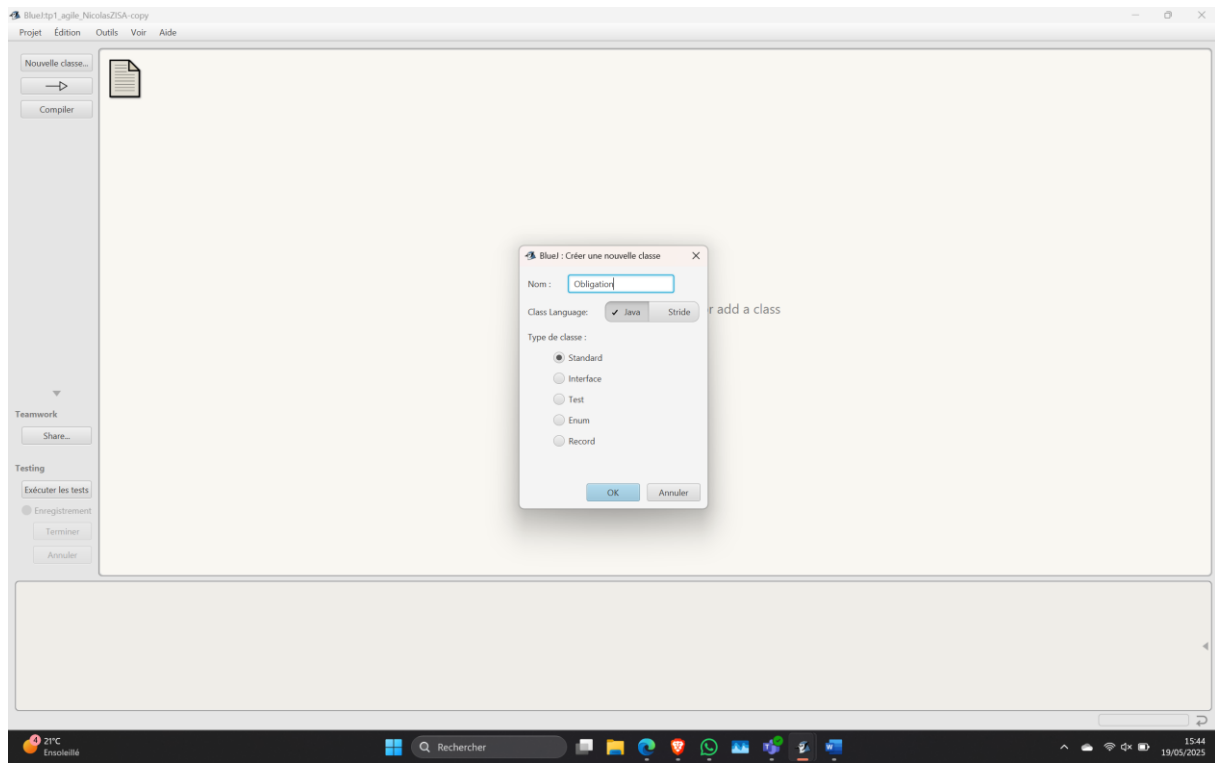
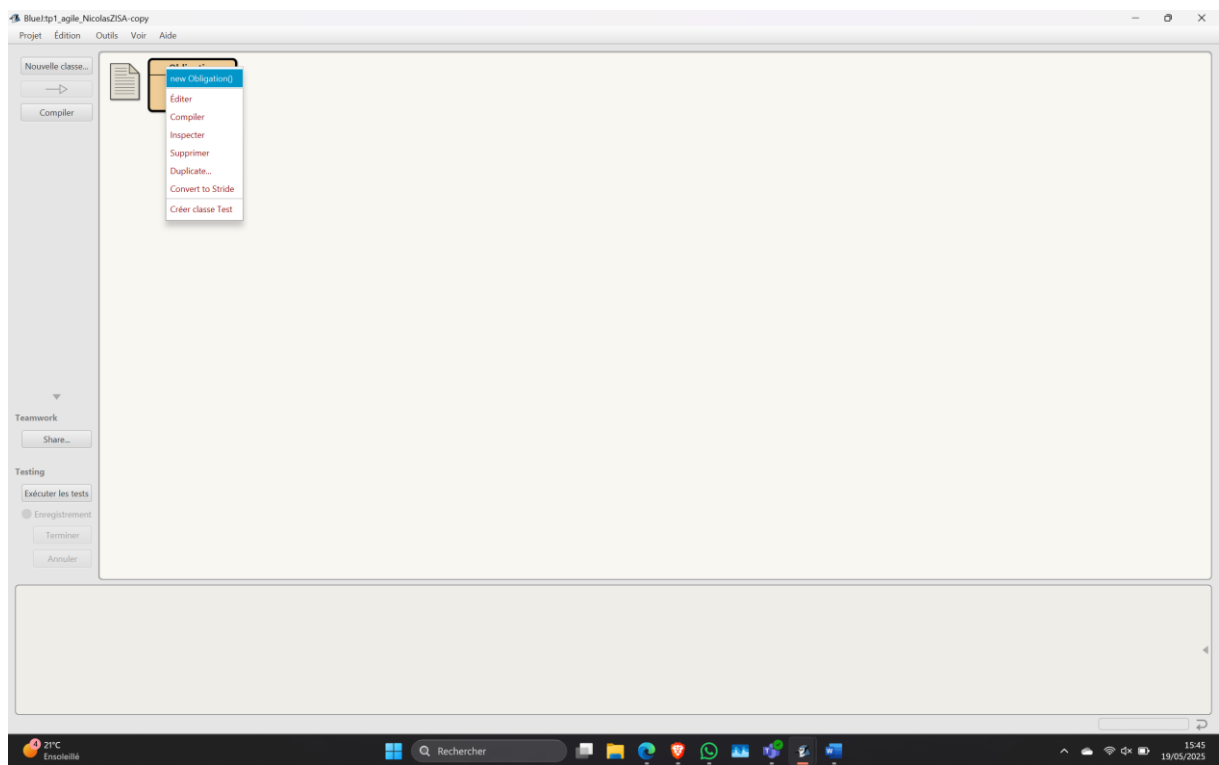
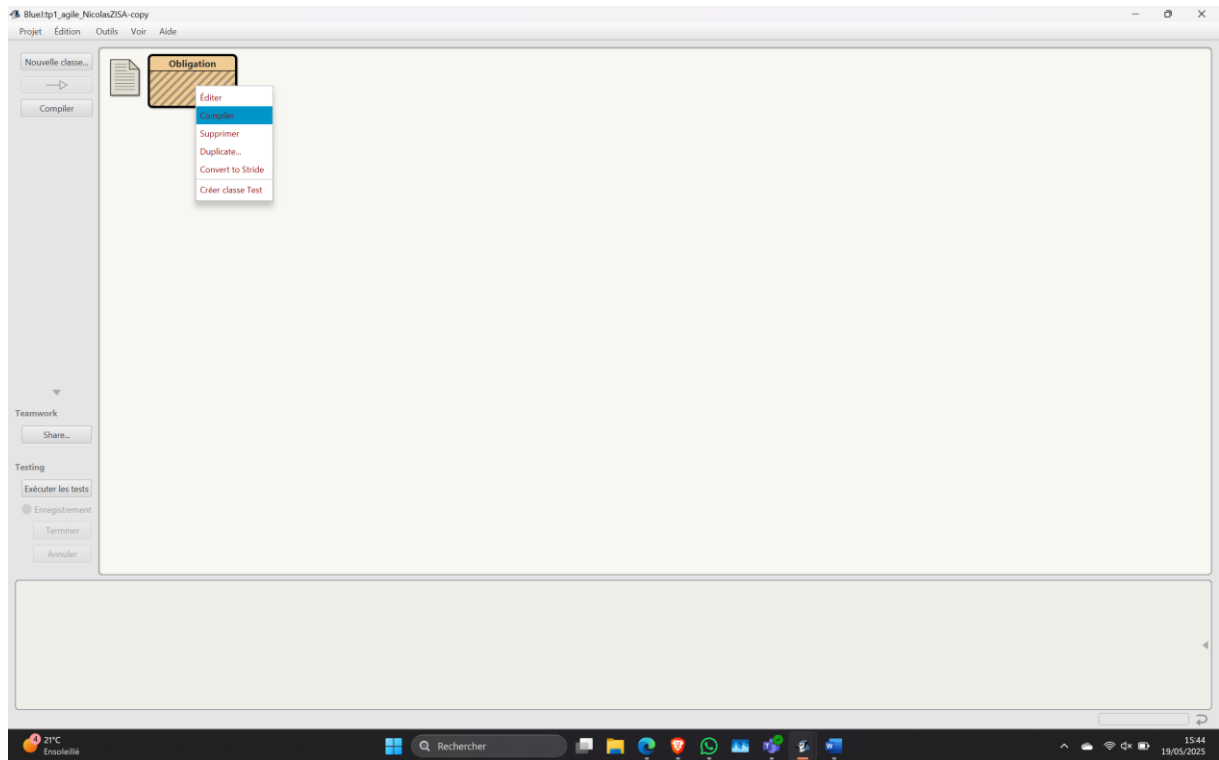


## Tutoriel : Manipulation de tests avec BlueJ et la programmation orientée objet

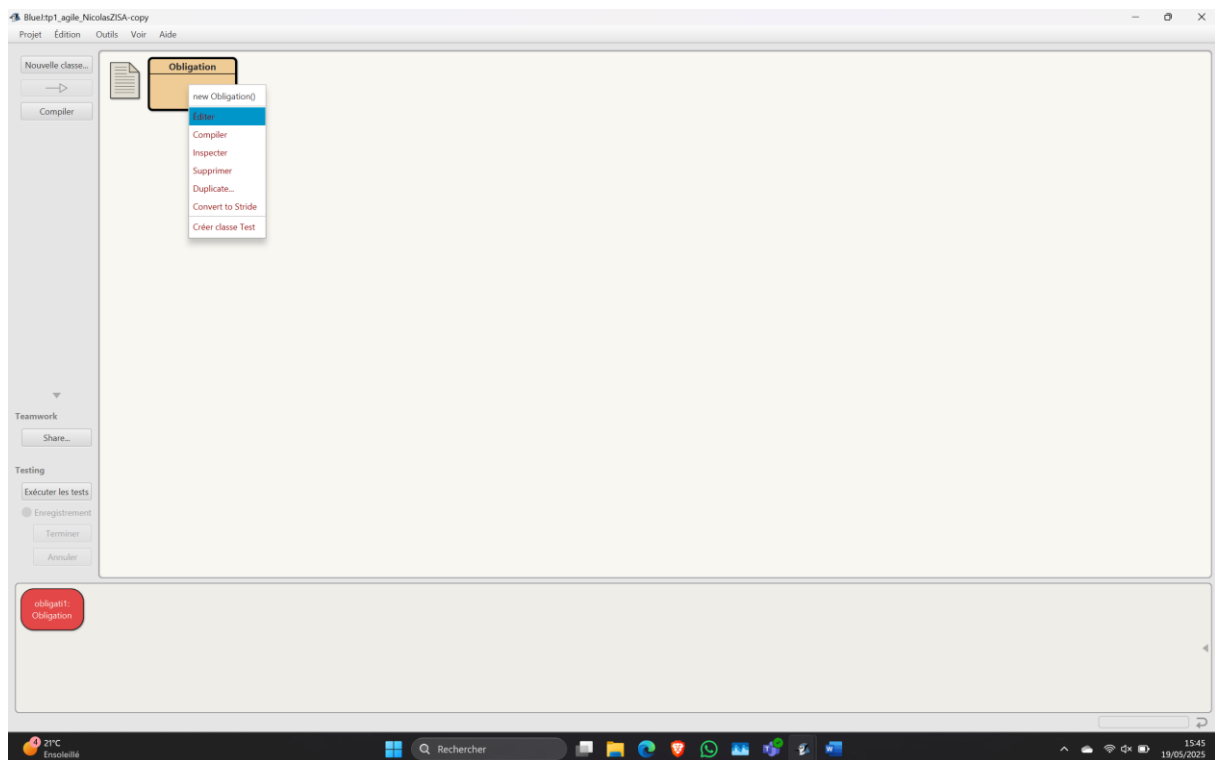
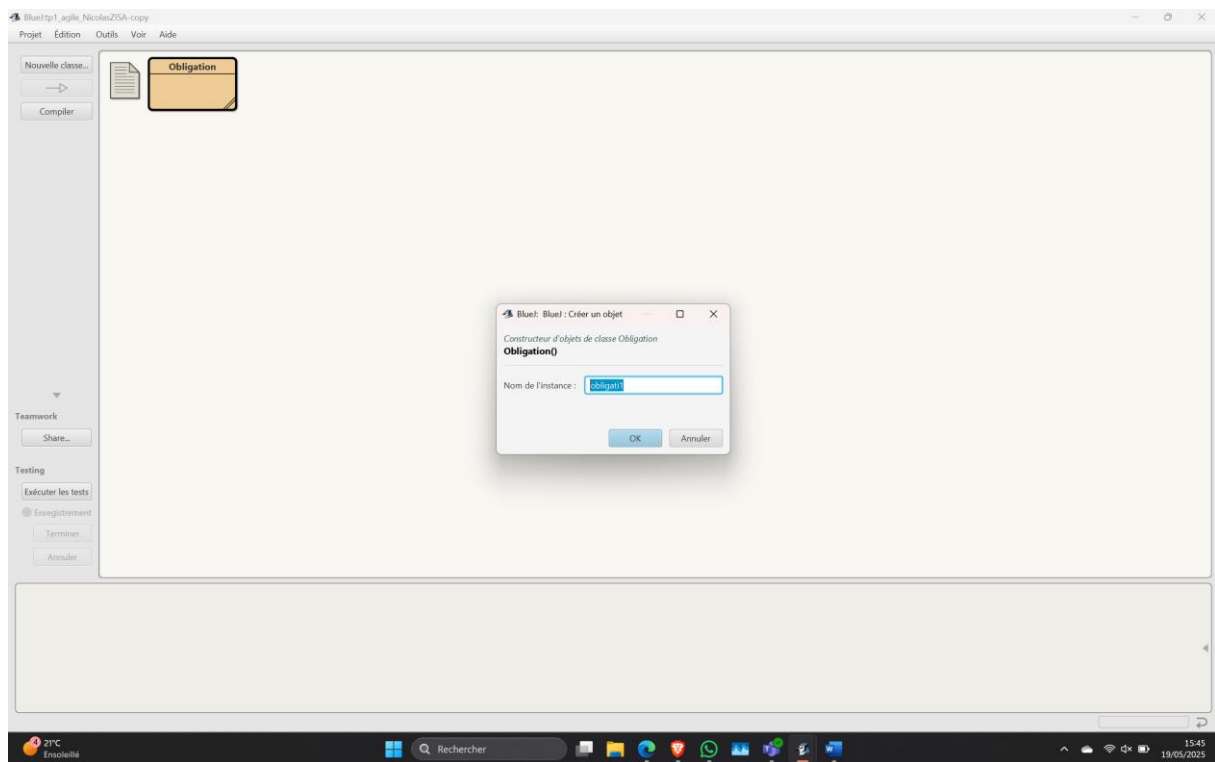
Nous commençons par créer une classe obligation qui représente un actif financier possédant une maturité, un nominal et un taux.

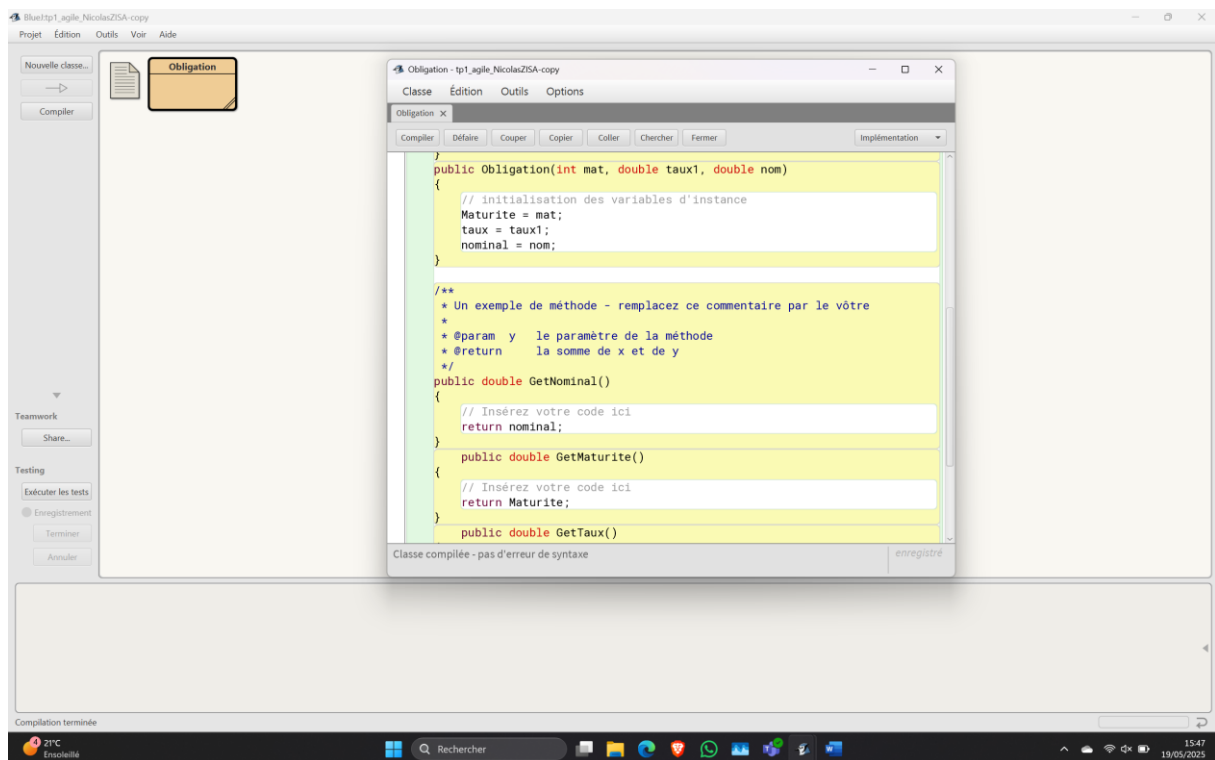
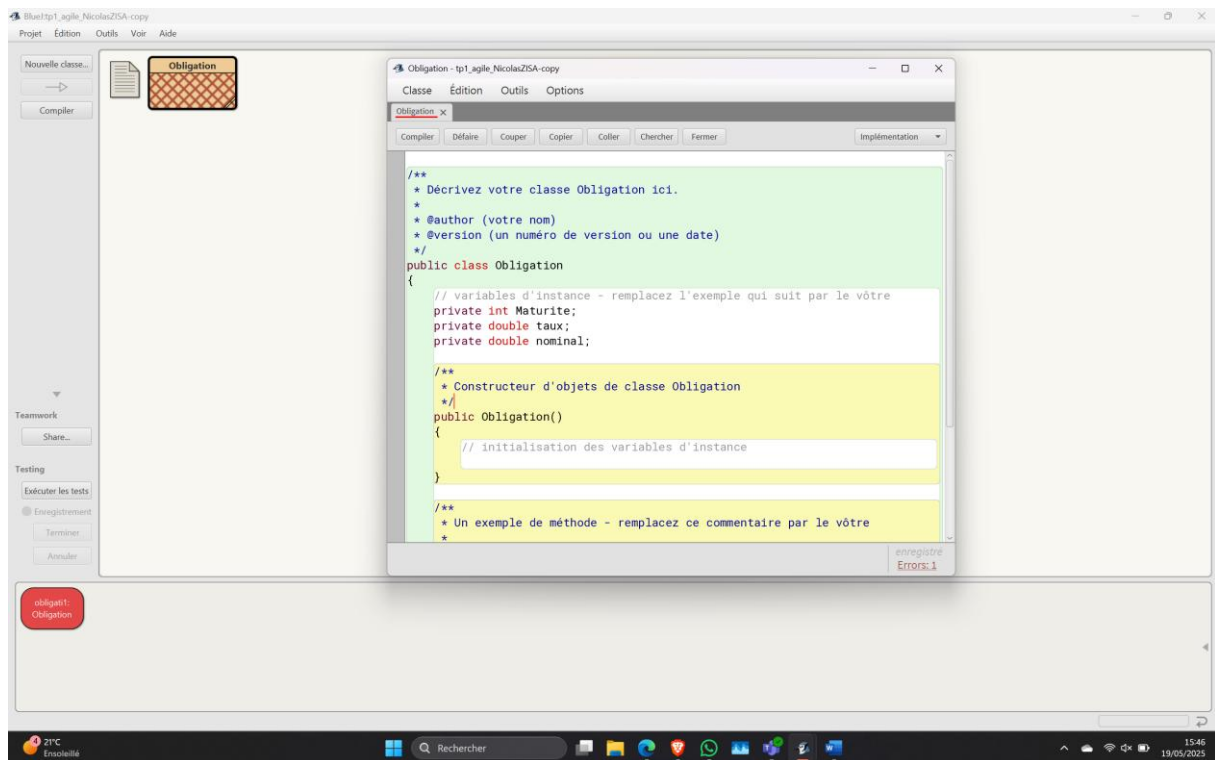


Nous pouvons la compiler et l'instancier même sans schéma clair de sa constitution.

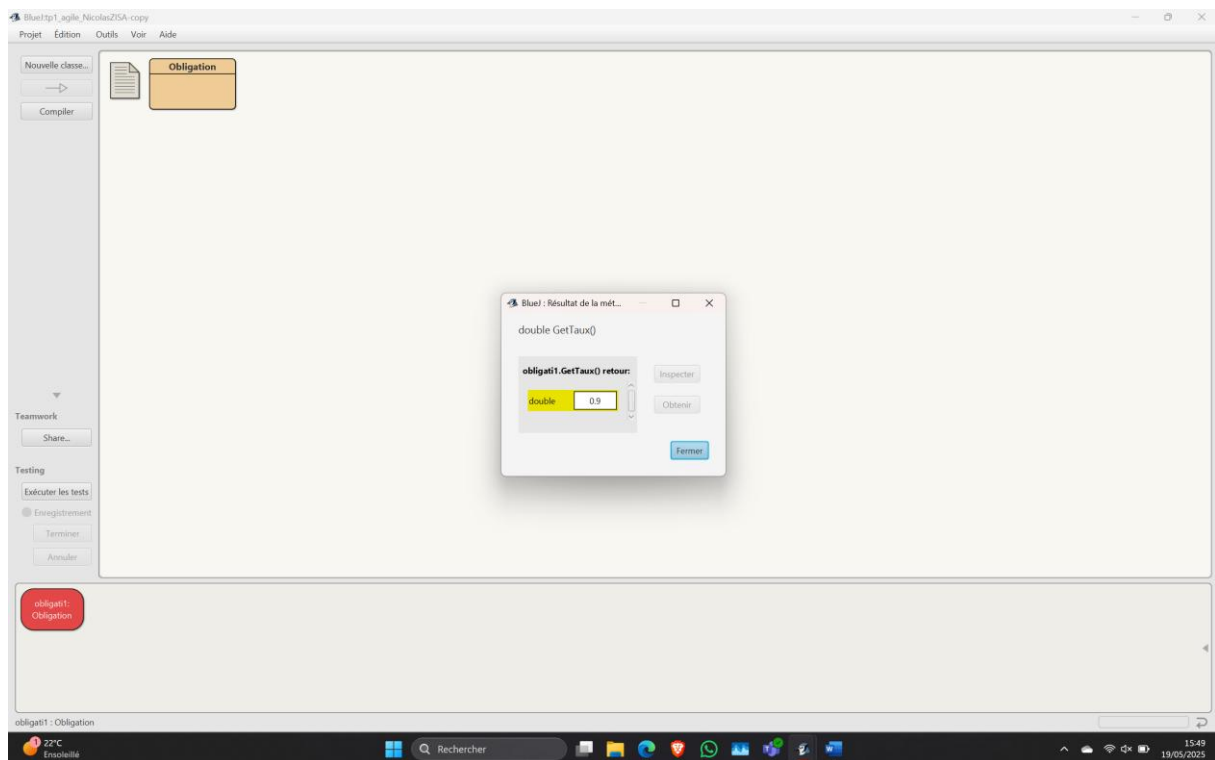
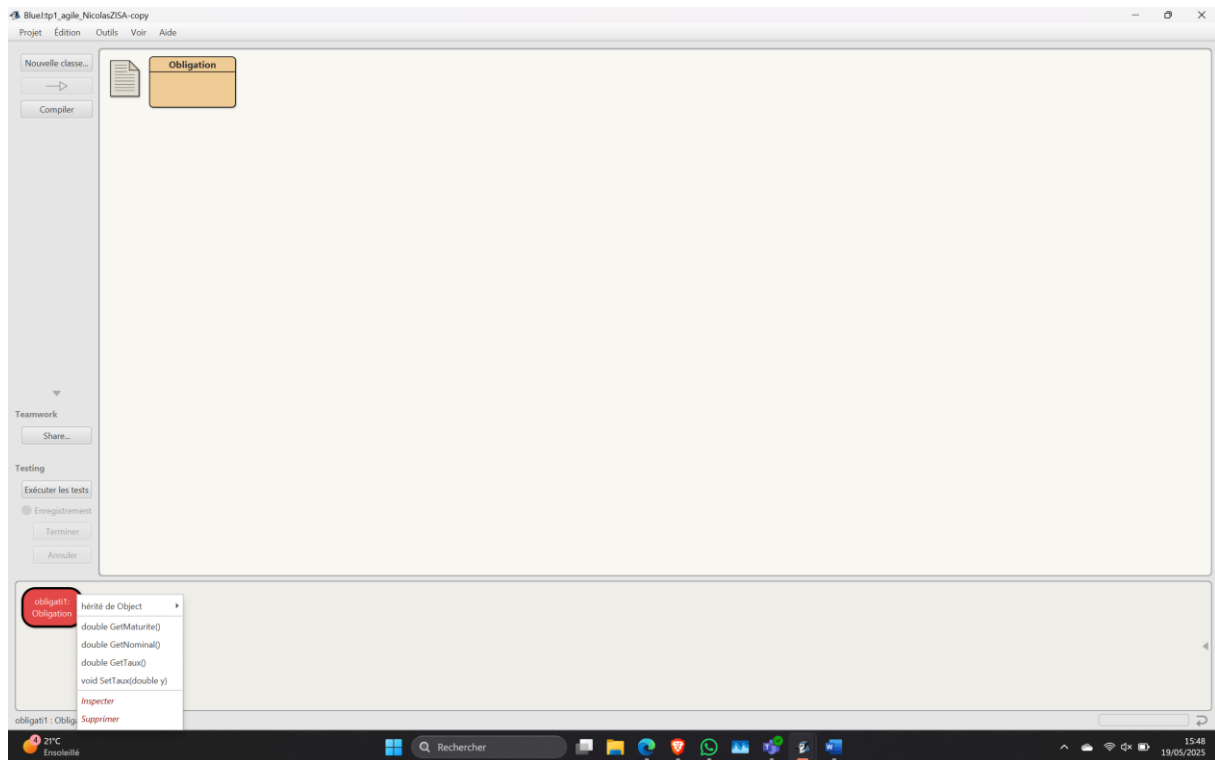


Commençons à mieux définir notre classe.

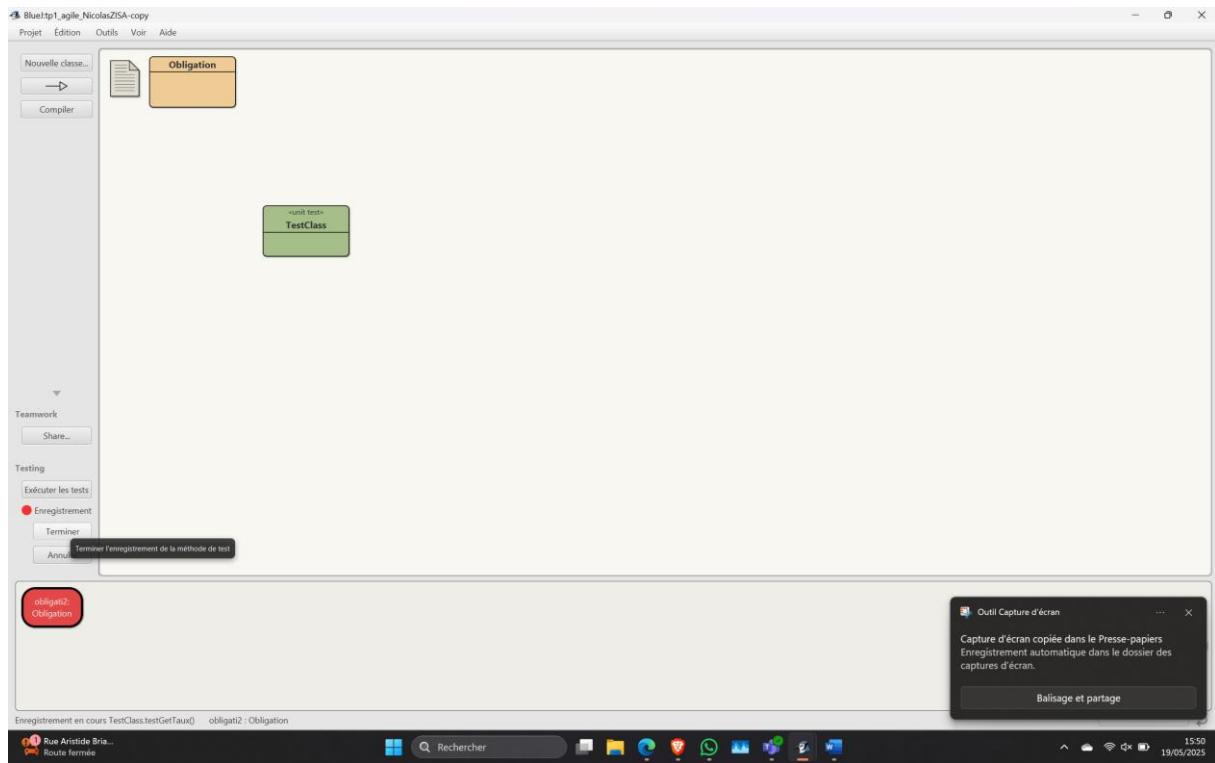




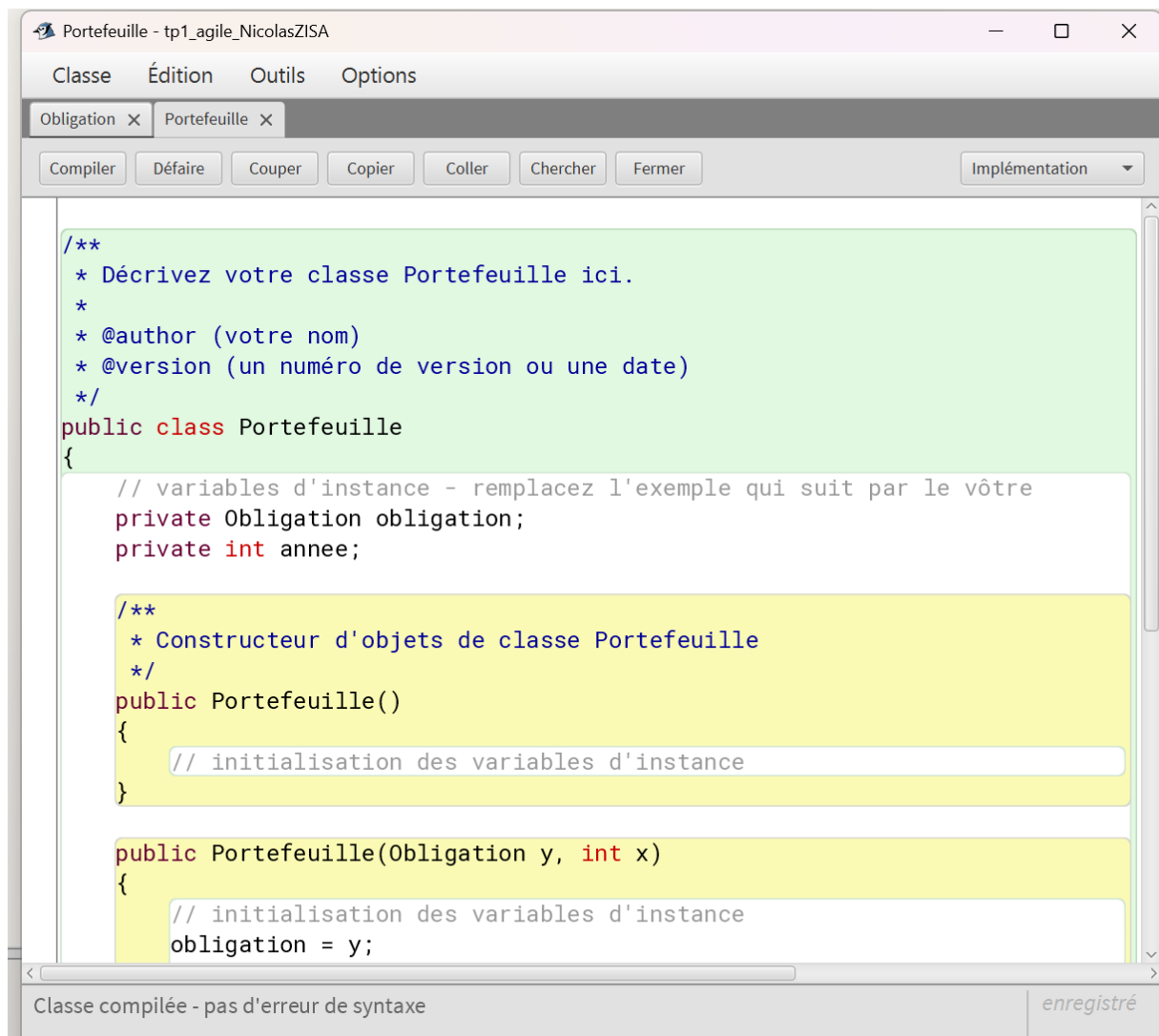
Instancions notre classe et essayons notre méthode GetTaux().



Maintenant pour notre premier test nous créons une classe test, nous la compilons, nous définissons notre première méthode de test pour la méthode GetTaux.



Maintenant nous créons une nouvelle classe Portefeuille, qui va contenir une obligation à un instant t.



The screenshot shows a Java IDE window titled "Portefeuille - tp1\_agile\_NicolasZISA". The window has a menu bar with "Classe", "Édition", "Outils", and "Options". Below the menu bar is a toolbar with buttons for "Compiler", "Défaire", "Couper", "Copier", "Coller", "Chercher", and "Fermer". A dropdown menu labeled "Implémentation" is on the right. The main editor area displays the following Java code:

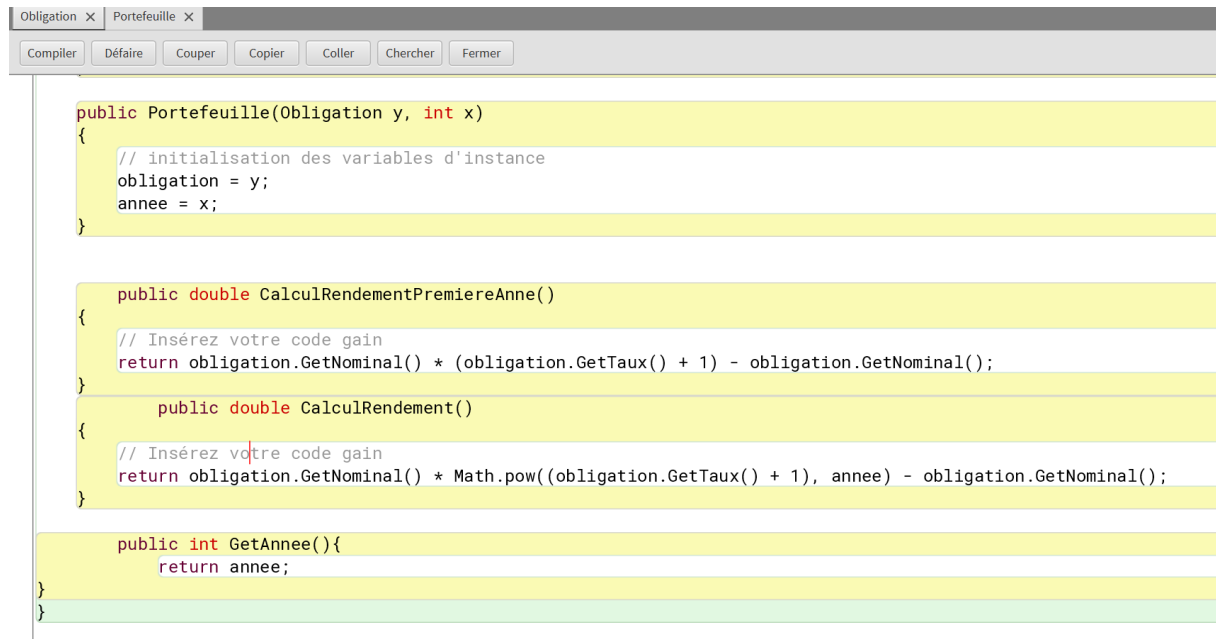
```
/**
 * Décrivez votre classe Portefeuille ici.
 *
 * @author (votre nom)
 * @version (un numéro de version ou une date)
 */
public class Portefeuille
{
    // variables d'instance - remplacez l'exemple qui suit par le vôtre
    private Obligation obligation;
    private int annee;

    /**
     * Constructeur d'objets de classe Portefeuille
     */
    public Portefeuille()
    {
        // initialisation des variables d'instance
    }

    public Portefeuille(Obligation y, int x)
    {
        // initialisation des variables d'instance
        obligation = y;
    }
}
```

At the bottom of the window, a status bar shows "Classe compilée - pas d'erreur de syntaxe" on the left and "enregistré" on the right.

Enfin, nous utilisons nos méthodes de la classe Obligation afin de calculer le rendement de notre portefeuille.



```
public Portefeuille(Obligation y, int x)
{
    // initialisation des variables d'instance
    obligation = y;
    annee = x;
}

public double CalculRendementPremiereAnne()
{
    // Insérez votre code gain
    return obligation.GetNominal() * (obligation.GetTaux() + 1) - obligation.GetNominal();
}

public double CalculRendement()
{
    // Insérez votre code gain
    return obligation.GetNominal() * Math.pow((obligation.GetTaux() + 1), annee) - obligation.GetNominal();
}

public int GetAnnee(){
    return annee;
}
```



«unit test»

**testObligation**

Tout tester

void testCapvcap()

void testFixture()

void testGetNominal()

void testfff()

Éditer

Compiler

Inspecter

Supprimer

Duplicate...

Enregistrer une méthode de test

Bureau Objets --> Engagements

Engagements --> Bureau Objets

BlueJ: BlueJ : Créer un objet

### Portefeuille(Obligation y, int x)

Nom de l'instance :

portefeu1

**new Portefeuille(**

obligati1

2

OK

Annuler

Nous pouvons maintenant créer une nouvelle méthode de test sur la méthode CalculRendement de la classe Portefeuille.

The screenshot shows the BlueJ IDE interface. The top menu bar includes 'Projet', 'Édition', 'Outils', 'Voir', and 'Aide'. On the left sidebar, there are buttons for 'Nouvelle classe...', a right-pointing arrow, and 'Compiler'. Below these are sections for 'Teamwork' (with a 'Share...' button) and 'Testing' (with 'Exécuter les tests', 'Enregistrement', 'Terminer', and 'Annuler' buttons). The main workspace displays a class diagram with three classes: 'Portefeuille', 'Obligation', and 'PortefeuilleTems'. 'Portefeuille' and 'Obligation' are connected by a bidirectional dashed arrow. 'PortefeuilleTems' has a dashed arrow pointing to 'Portefeuille'. A green box labeled '«unit test» testObligation' is positioned below the 'Obligation' class. A context menu is open over this box, listing options: 'Tout tester', 'void testCapvcap()', 'void testFixture()', 'void testGetNominal()', 'void testffff()', 'Éditer', 'Compiler', 'Inspector', 'Supprimer', 'Duplicate...', 'Enregistrer une méthode de test' (highlighted in blue), 'Bureau Objets --> Engagements', and 'Engagements --> Bureau Objets'. At the bottom, there are two red buttons: 'obligati1: Obligation' and 'portefeu1: Portefeuille'.

BlueJ:tp1\_agile\_NicolasZISA

ProjetÉditionOutilsVoirAide

Nouvelle classe...  
→  
Compiler

Portefeuille

PortefeuilleTemps

Obligation

«unit test»  
testObligation

Teamwork

Share...

Testing

Exécuter les tests

● Enregistrement

Terminer

Annuler

obligati1:  
Obligation

portefeu1:  
Portefeuille

Enregistrer une méthode de test

Spécifiez un nom pour cette méthode de test. Son enregistrement débutera ensuite.

testfix

OKAnnuler

BlueJ:tp1\_agile\_NicolasZISA

ProjetÉditionOutilsVoirAide

Nouvelle classe...

→

Compiler

Teamwork

Share...

Testing

Exécuter les tests

● Enregistrement

Terminer

Annuler

Portefeuille

Portefeuille

PortefeuilleTems

Obligation

«unit test»  
testObligation

obligati3:  
Obligation

portefe:  
Portefe

hérité de Object

double CalculRendement()

double CalculRendementPremiereAnne()

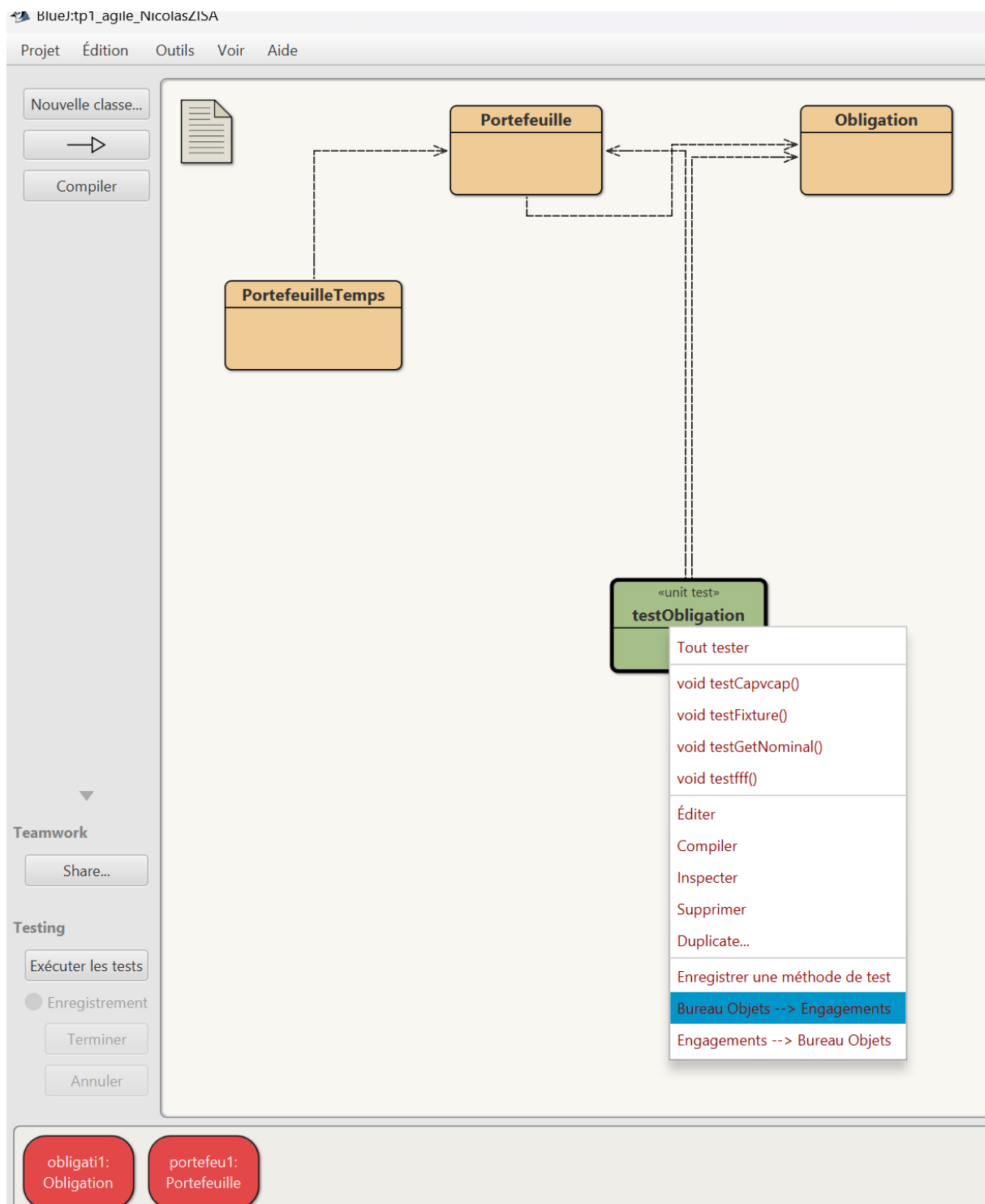
int GetAnnee()

Inspector

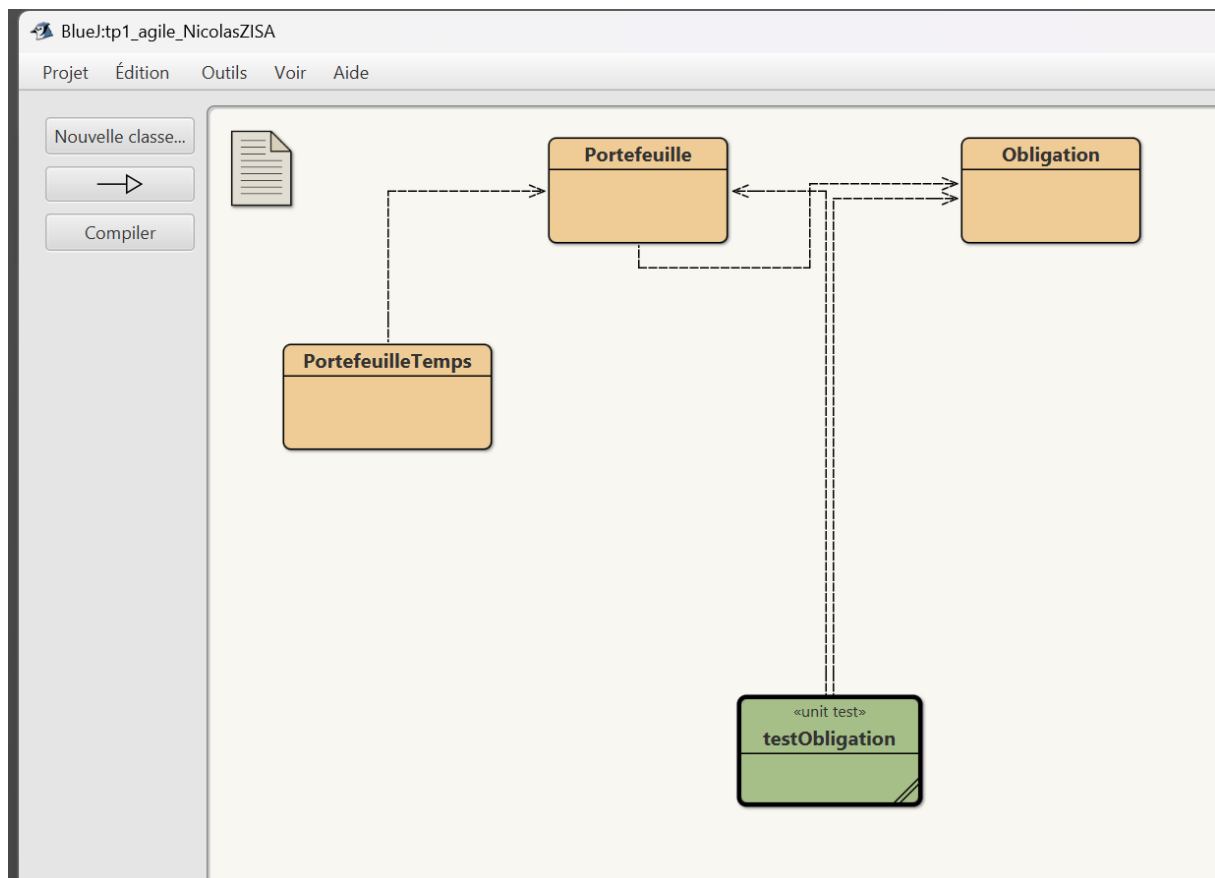
Supprimer

Enregistrement en cours testOb

Maintenant nous pouvons sauvegarder nos instances dans la fixture de la classe test.



Nous allons créer une classe PortefeuilleTemps, constituée d'une collection de portefeuille. Tous contiennent la même obligation mais à des instants différents.



```
import java.util.ArrayList;

/**
 * Décrivez votre classe PortefeuilleTemps ici.
 *
 * @author (votre nom)
 * @version (un numéro de version ou une date)
 */
public class PortefeuilleTemps
{
    // variables d'instance - remplacez l'exemple qui suit par le vôtre
    private ArrayList Set_Portefeuille;

    /**
     * Constructeur d'objets de classe PortefeuilleTemps
     */
    public PortefeuilleTemps()
    {
        // initialisation des variables d'instance
        Set_Portefeuille = new ArrayList();
    }

    /**
     * Un exemple de méthode - remplacez ce commentaire par le vôtre
     *
     * @param y le paramètre de la méthode
     * @return la somme de x et de y
     */
    public void Add(Portefeuille y)
    {
        // Insérez votre code ici
        Set_Portefeuille.add(y);
    }
}
```