TUTO: PANIER ACHAT

Contexte du projet :

Le projet consiste à modéliser un système de panier d'achat en Java, avec une interaction simple entre deux classes principales : PanierAchat et Article. L'objectif est de créer un programme permettant de gérer les articles ajoutés à un panier, de calculer le montant total du panier, et d'appliquer des remises selon des pourcentages donnés.

Classes principales du projet :

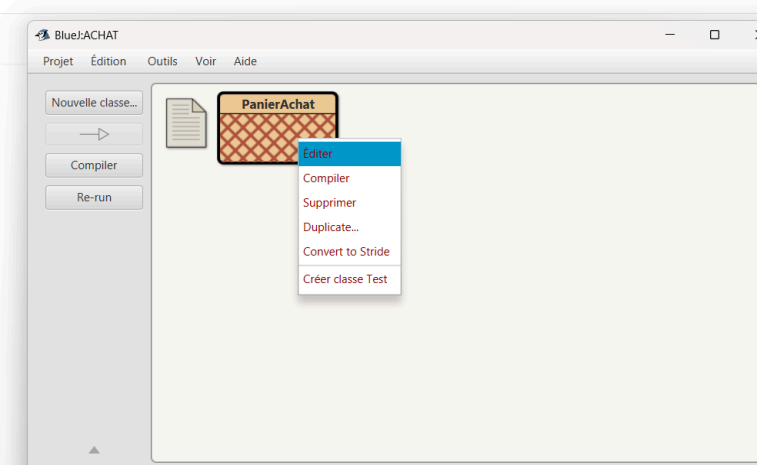
1. **PanierAchat**
   Cette classe représente le panier d'achat. Elle contient des attributs tels que la liste des articles et le montant total. Elle est responsable de l'ajout d'articles au panier et du calcul du montant total après application d'éventuelles remises.

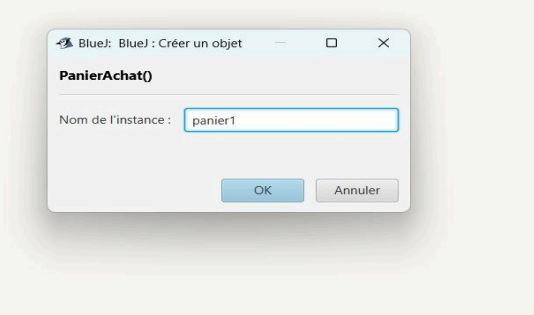
2. **Article**
   Cette classe représente un article individuel du panier. Chaque article possède un nom et un prix. Ces objets sont utilisés par la classe PanierAchat pour modéliser les articles contenus dans un panier d'achat.

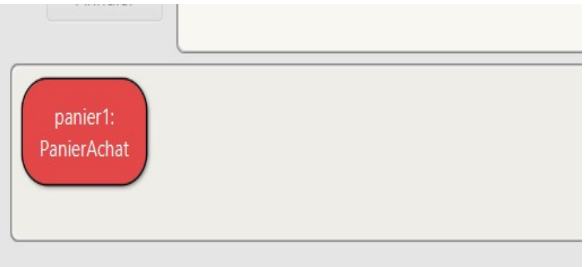
3. **PanierAchatTest**
   Il s'agit de la classe de test unitaire. Elle permet de tester le bon fonctionnement des méthodes de la classe PanierAchat, en vérifiant par exemple si le nombre d'articles et le montant total sont corrects après avoir ajouté des articles au panier.

```java
import java.util.ArrayList;

public class PanierAchat {

    // ✅ 1. ATTRIBUTS
    private ArrayList<Article> articles;
    private double montantTotal;

    // ✅ 2. CONSTRUCTEUR
    public PanierAchat() {
        articles = new ArrayList<>();
        montantTotal = 0;
    }

    // ✅ 3. ACCESSEURS (getters)
    public double getMontantTotal() {
        return montantTotal;
    }

    public int getNombreArticles() {
        return articles.size();
    }
```

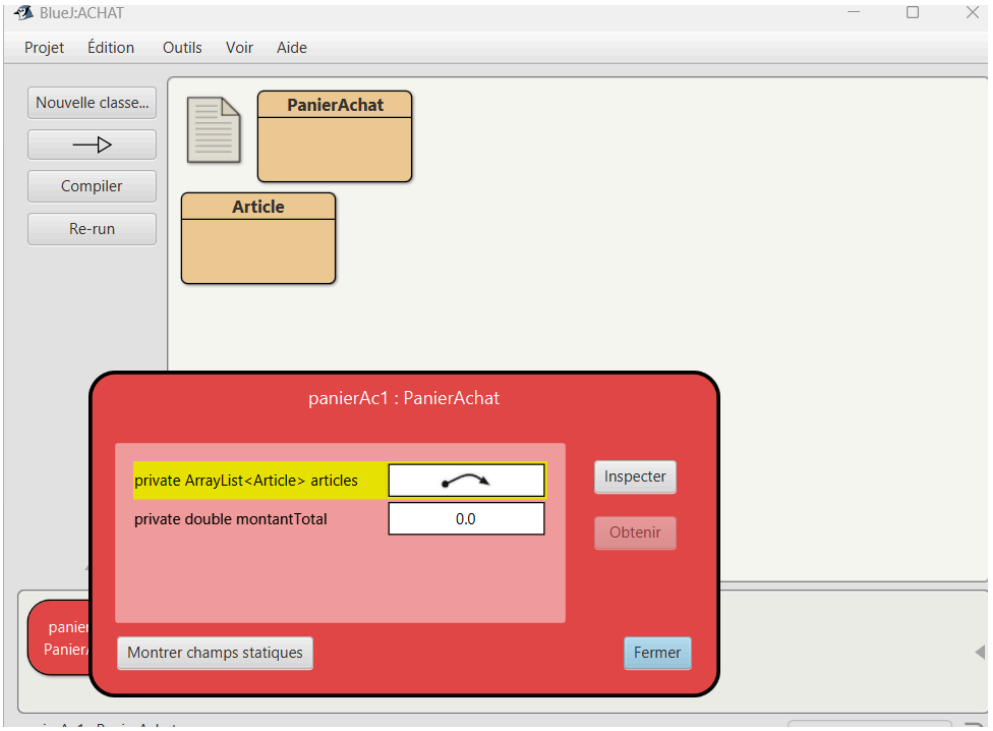
- On crée la classe PanierAchat, qui devient la classe principale du projet.
- On définit le constructeur PanierAchat(), chargé d'initialiser l'objet.
- On ajoute deux attributs :
  articles
  montant Total
- On met également en place les accesseurs associés pour pouvoir consulter l'état de l'objet.
  Ensuite ,on compile la classe PanierAchat via le menu contextuel de BlueJ.
- La compilation permet de vérifier que la classe et son constructeur PanierAchat() sont valides.



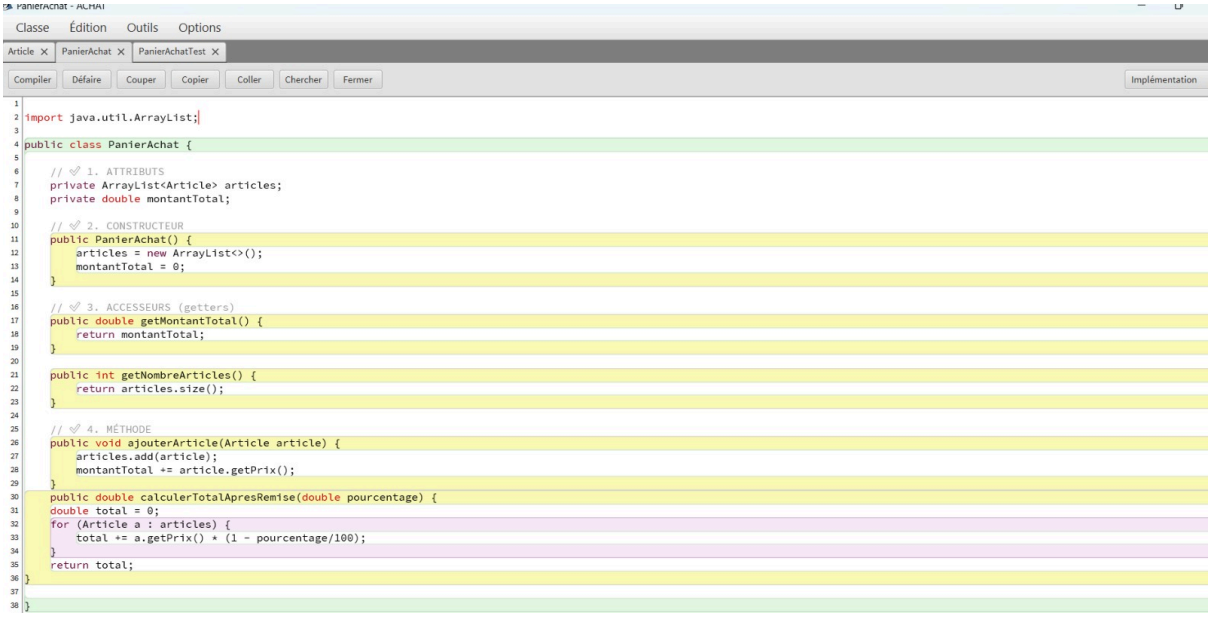




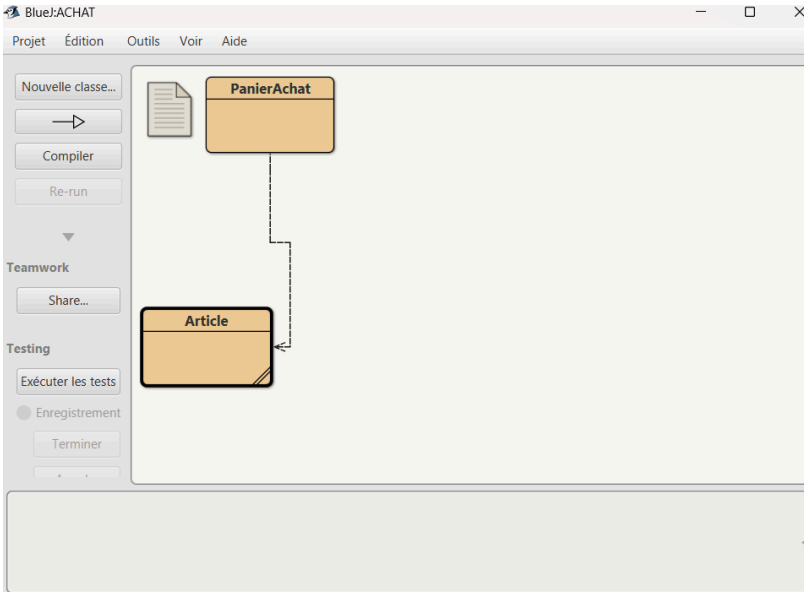
- On crée un objet nommé panier1 à partir de la classe fétiche.
  L'objet panier1 : PanierAchat apparaît dans l'Object Bench.
- Cela confirme que le constructeur PanierAchat() a bien été exécuté et que l'objet existe.
- On ajoute une seconde classe nommée Article au projet.
  Cette classe est associée de manière unidirectionnelle à la classe fétiche PanierAchat, via l'attribut articles.



- On inspecte l'objet panier1 pour visualiser ses attributs :
  articles (initialisé par le constructeur)
  montantTotal (initialisé à 0.0)
- Cette inspection permet de constater l'état de l'objet après instanciation et après l'exécution du constructeur PanierAchat().

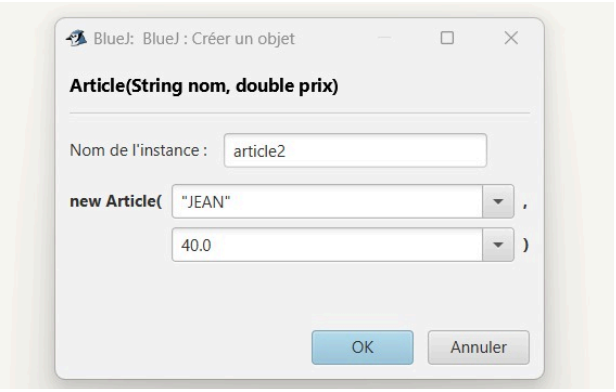


- On a ajouté 2 méthodes:
  Une méthode ajouterArticle(Article article) est ajoutée pour modifier l'état de l'objet en ajoutant un article au panier.
  Une méthode calculerTotalApresRemise(double pourcentage) est définie pour collaborer avec les objets Article afin de produire un résultat basé sur l'ensemble des articles du panier.

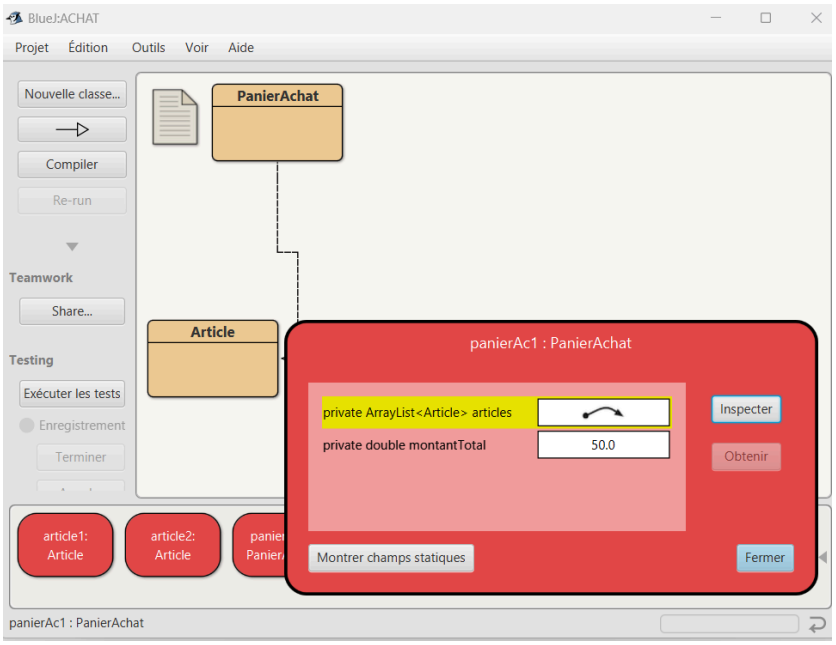


- On visualise :
  La classe PanierAchat est la classe fétiche du projet.
  La classe Article est ajoutée comme seconde classe.
  Une association unidirectionnelle est établie entre PanierAchat et Article.

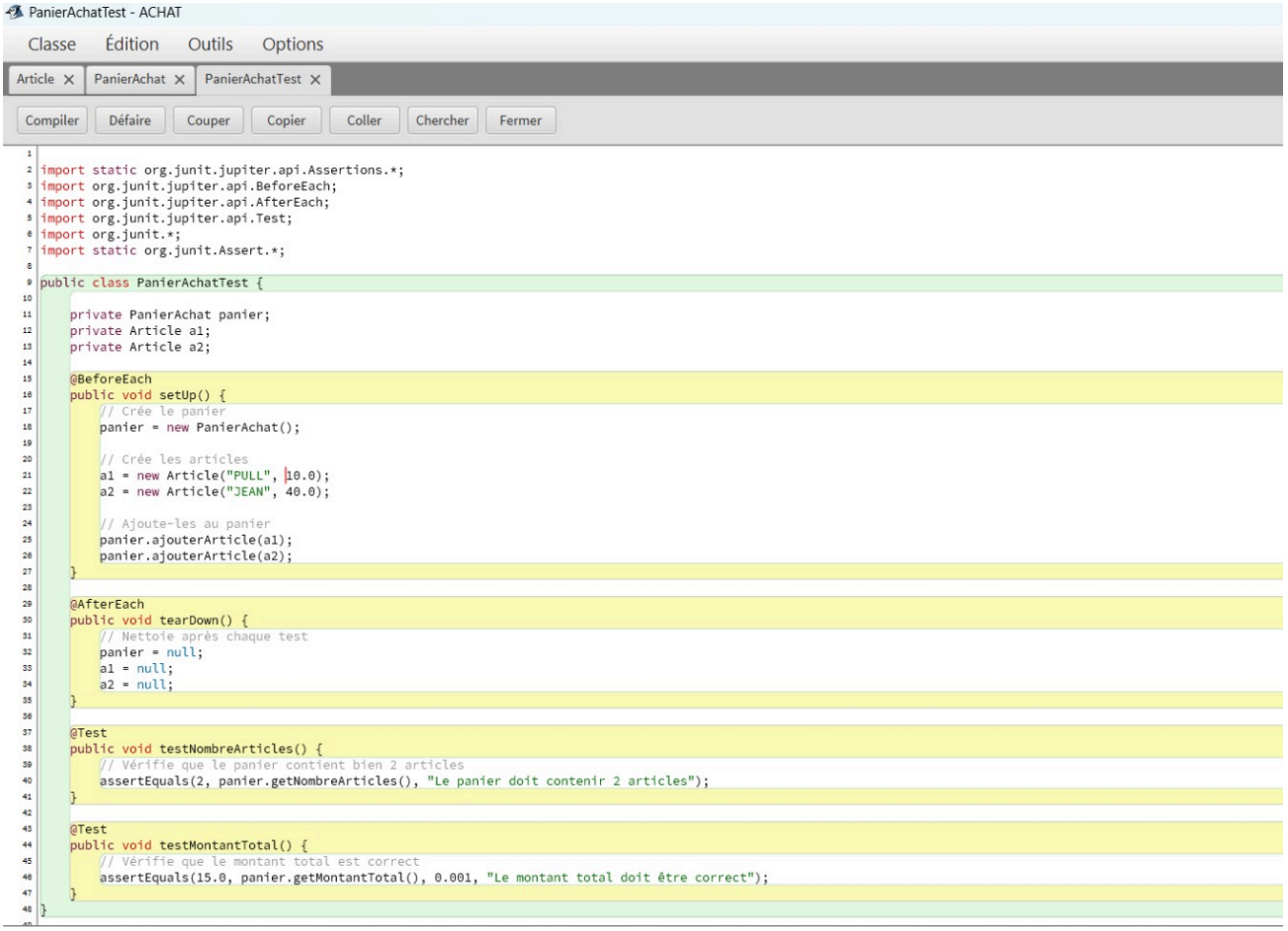
Cette association correspond à l'utilisation de la classe Article par PanierAchat, notamment via l'attribut articles



- On instancie deux objets Article en appelant le constructeur Article(String nom, double prix) :
  article1 avec le nom "PULL" et le prix 10.0
  article2 avec le nom "JEAN" et le prix 40.0
- Ces objets Article sont ensuite utilisés par l'objet panierAc1 : PanierAchat, créé précédemment via le constructeur PanierAchat().

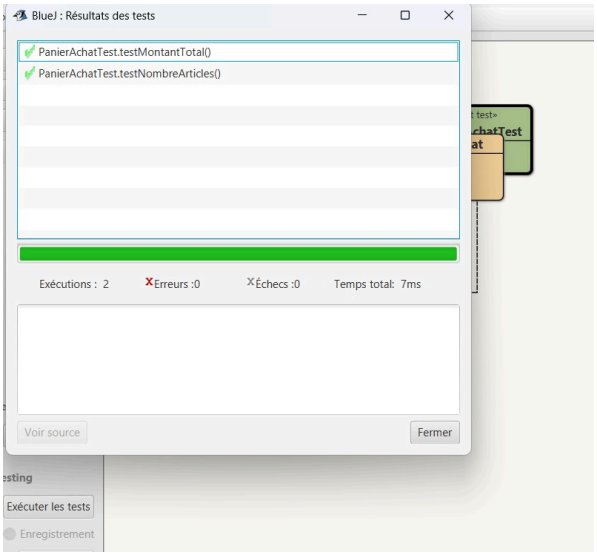


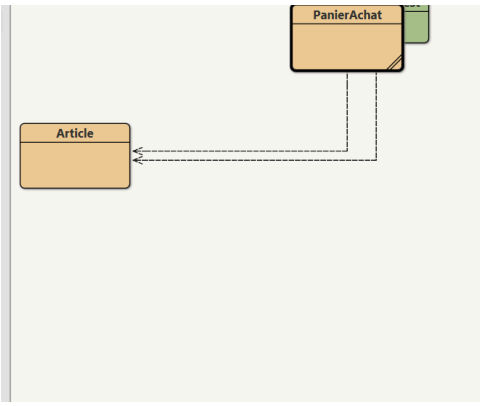
- On inspecte l'objet panierAc1 et on observe :
  l'attribut articles, qui contient maintenant les objets article1 et article2,
  l'attribut montantTotal, dont la valeur est passée à 50.0, ce qui reflète l'effet des méthodes appelées sur le panier.




```java
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.Test;
import org.junit.*;
import static org.junit.Assert.*;

public class PanierAchatTest {

    private PanierAchat panier;
    private Article a1;
    private Article a2;

    @BeforeEach
    public void setUp() {
        // Crée le panier
        panier = new PanierAchat();

        // Crée les articles
        a1 = new Article("PULL", 10.0);
        a2 = new Article("JEAN", 40.0);

        // Ajoute-les au panier
        panier.ajouterArticle(a1);
        panier.ajouterArticle(a2);
    }

    @AfterEach
    public void tearDown() {
        // Nettoie après chaque test
        panier = null;
        a1 = null;
        a2 = null;
    }

    @Test
    public void testNombreArticles() {
        // Vérifie que le panier contient bien 2 articles
        assertEquals(2, panier.getNombreArticles(), "Le panier doit contenir 2 articles");
    }

    @Test
    public void testMontantTotal() {
        // Vérifie que le montant total est correct
        assertEquals(15.0, panier.getMontantTotal(), 0.001, "Le montant total doit être correct");
    }
}
```

- On crée la classe de test PanierAchatTest associée à la classe PanierAchat.
  Une fixture est définie avec les attributs panier, a1 et a2.
  La méthode setUp() instancie le panier avec le constructeur PanierAchat(), crée deux articles avec le constructeur Article(String nom, double prix), puis les ajoute au panier.
  La méthode tearDown() remet les objets à null après chaque test





- On exécute ensuite les tests unitaires :
  testNombreArticles() vérifie l'état du panier après ajout des articles.
  testMontantTotal() vérifie le résultat obtenu après l'utilisation des méthodes du panier.
  Les résultats des tests s'affichent dans BlueJ avec une barre verte, indiquant que tous les tests ont réussi, sans erreurs ni échecs.
  Enfin, le diagramme de classes montre la relation entre PanierAchatTest, PanierAchat et Article, confirmant que la classe de test utilise les deux classes pour valider leur collaboration.