Formation Orienté Objet et Java de base

Atelier 3 : Les accesseurs

Encadré par M.BOULCHAHOUB

Objectif de l'atelier

L'objectif de cet atelier est de comprendre l'utilité des accesseurs (Les Getters et les Setters).

.

Vous allez apprendre à:

- *La problématique résolue par les accesseurs*
- *Les normes des accesseurs*
- *L'utilité des accesseurs*
- *La création et la génération automatique des accesseurs.*

Problématique de la consultation d'un attribut

Essayons tout d'abord de comprendre la problématique résolue par les accesseurs.
Créer un projet Java avec le nom TPAccesseurs

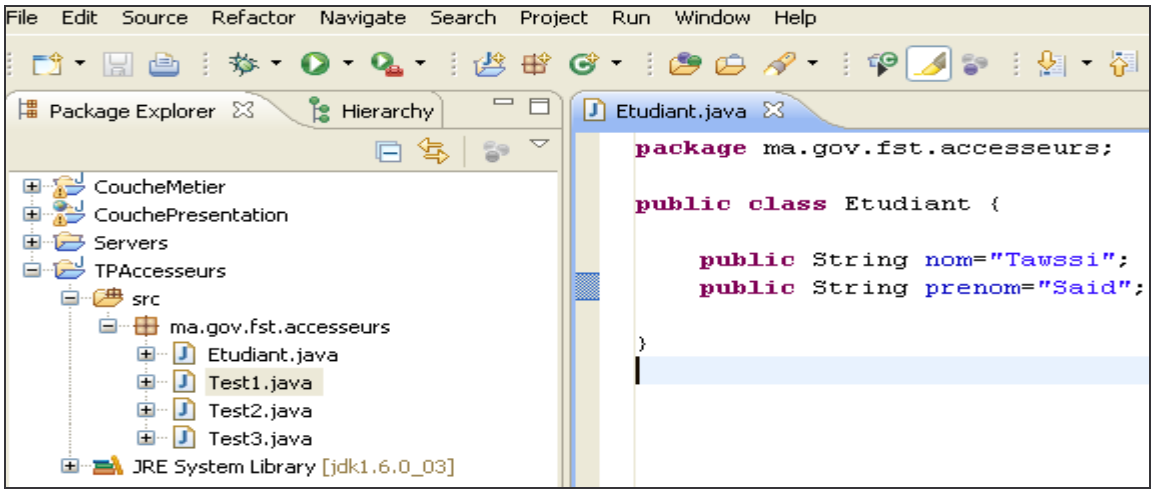
Créer un package ma.gov.fst.accesseurs

Créer une classe Etudiant avec deux attributs publics, le nom et le prénom.

Formation Orienté Objet et Java de base

Atelier 3 : Les accesseurs

Encadré par M.BOULCHAHOUB

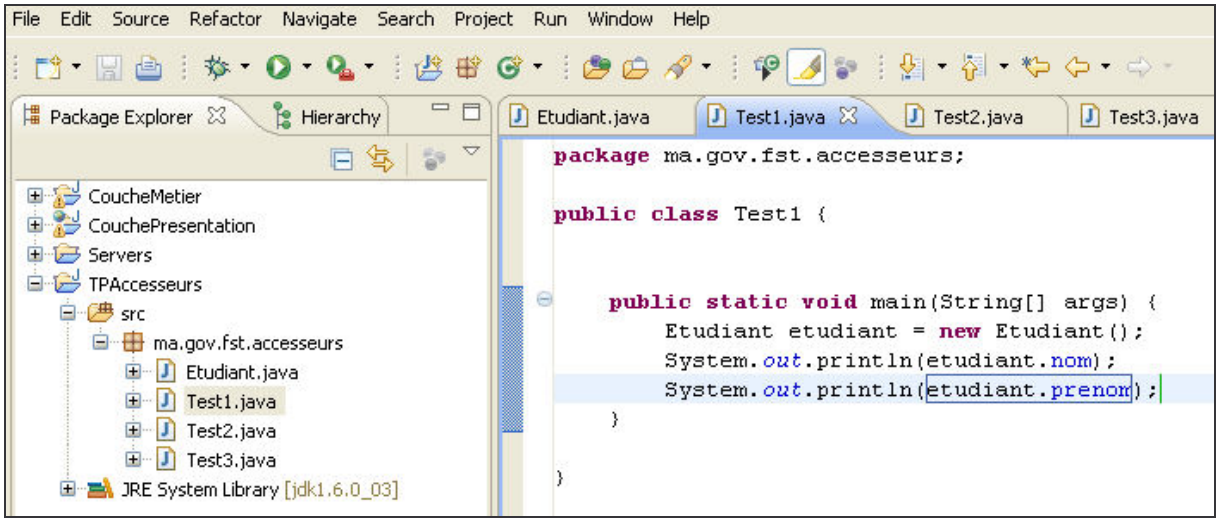


Créer trois autres classes contenant la méthode main. Ces classes portent le nom Test1, Test2 et Test3 (dans la suite de cet Atelier ces classes seront référencées par Testx)

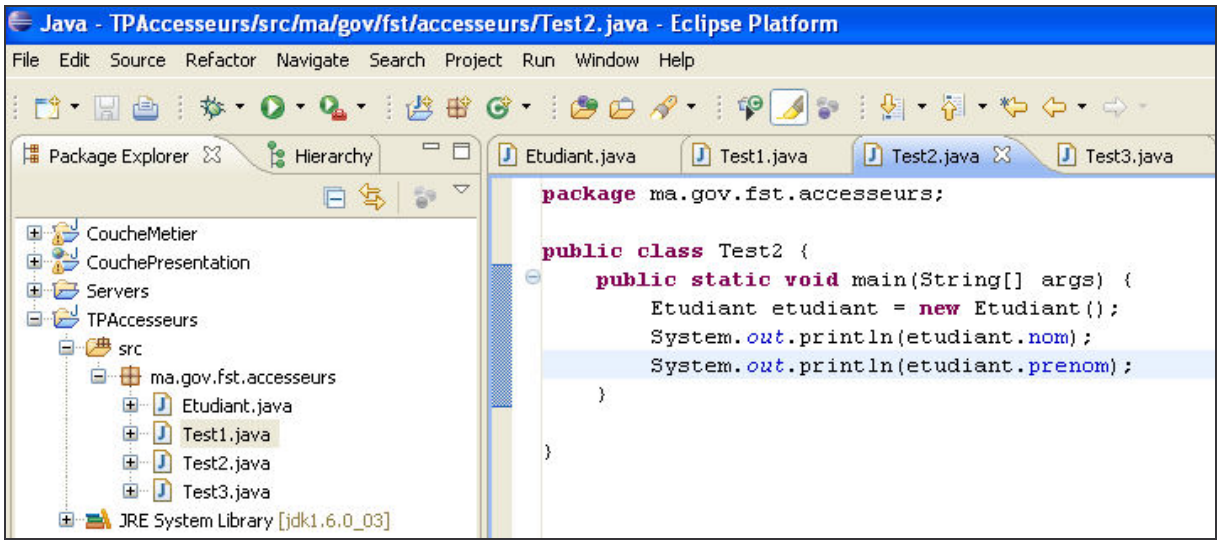
Dans la méthode main des classes Test1, Test2 et Test3, Instancier un objet de la classe Etudiant en lui donnant le nom etudiant.

Afficher le nom et le prénom de l'objet etudiant.

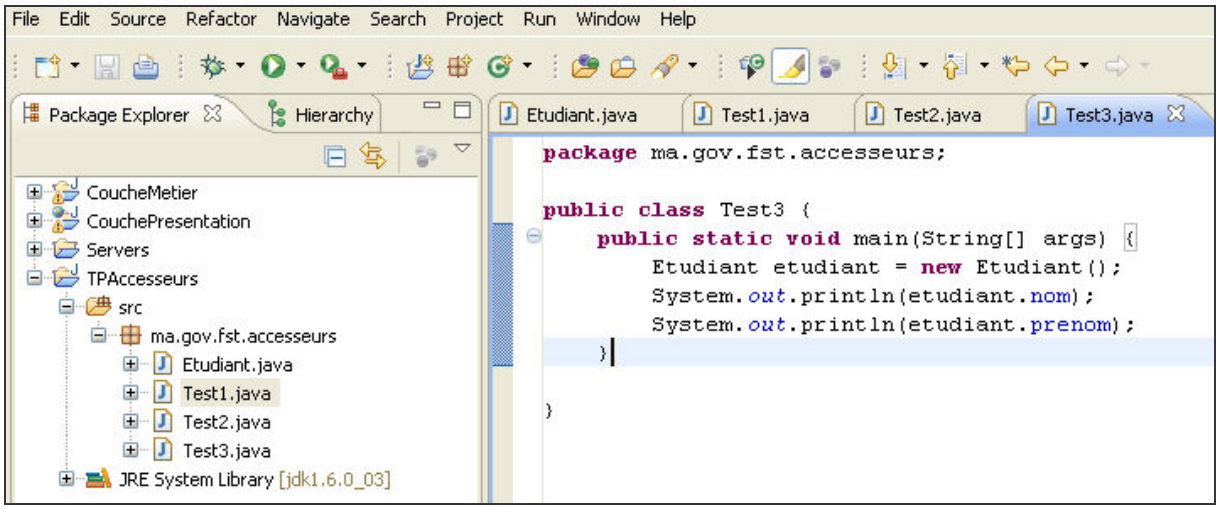
Classe Test1 :



Classe Test2 :



Classe Test3 :

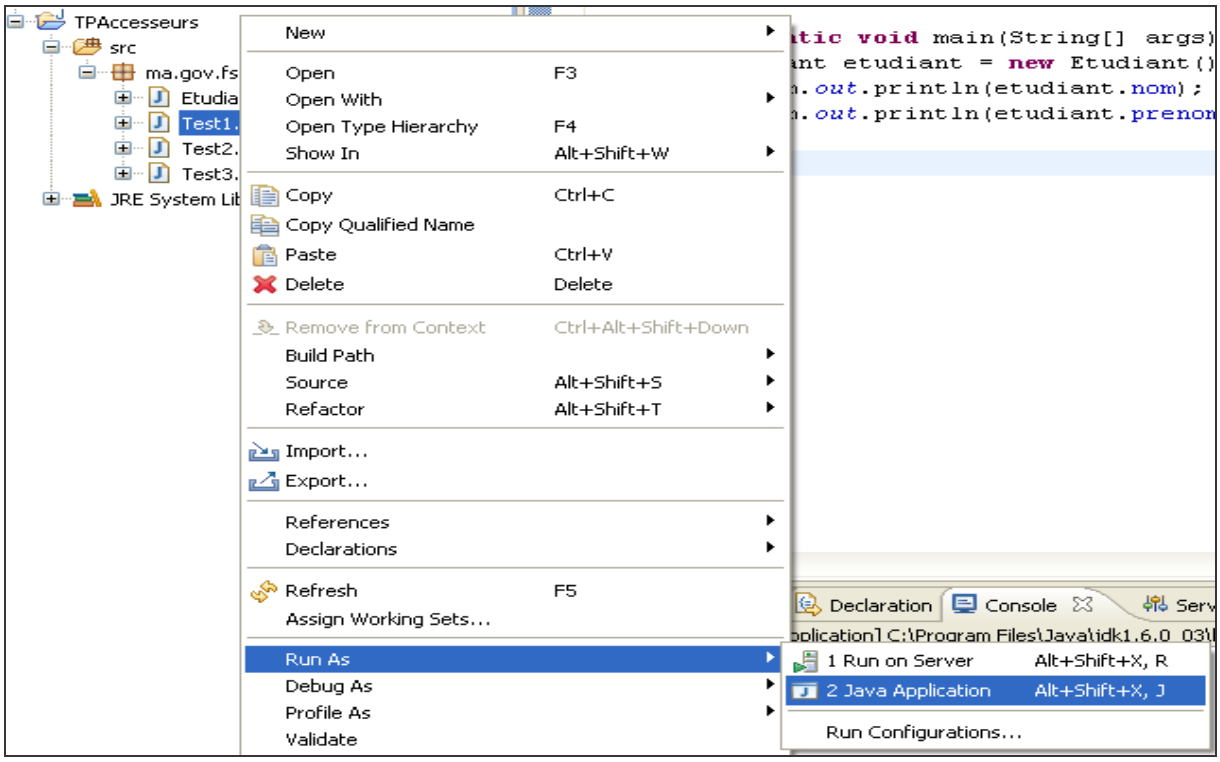


Exécuter les classes Test1, Test2 et Test3 :
Pour le faire Cliquer droit sur la classe Testx>run as>application java

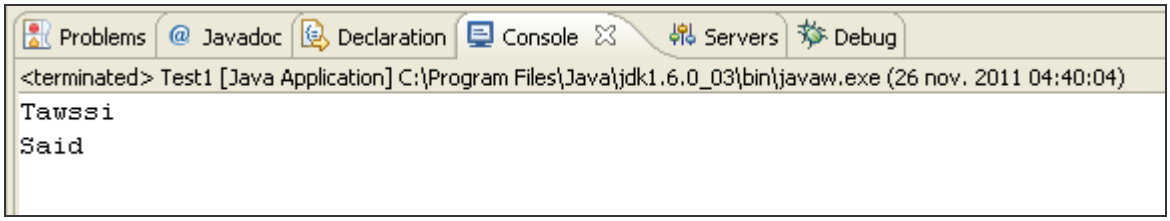
Formation Orienté Objet et Java de base

Atelier 3 : Les accesseurs

Encadré par M.BOULCHAHOUB



Vous devriez avoir dans la console :



Maintenant au moment de l'affichage, nous souhaitons faire précéder le nom par la chaine « Le nom est : »
Faire précéder également le prénom par « Le prénom est : »

Nous sommes obligés de modifier dans les trois classes Test1, Test2 et Test3 l'affichage comme suivant :

Pour la classe Test1

```
public class Test1 {
	public static void main(String[] args) {
		Etudiant etudiant = new Etudiant();
		System.out.println("Le nom est : "+etudiant.nom);
```

```
            System.out.println("Le prenom est : "+etudiant.prenom);
      }
}
```

Pour la classe Test2

```
public class Test2 {
      public static void main(String[] args) {
            Etudiant etudiant = new Etudiant();
            System.out.println("Le nom est : "+etudiant.nom);
            System.out.println("Le prenom est : "+etudiant.prenom);
      }
}
```

Pour la classe Test3

```
public class Test3 {
      public static void main(String[] args) {
            Etudiant etudiant = new Etudiant();
            System.out.println("Le nom est : "+etudiant.nom);
            System.out.println("Le prenom est : "+etudiant.prenom);
      }
}
```

Conclusion

Pour changer l'affichage, nous sommes obligés de modifier dans toutes les classes Testx. Imaginer la charge de travail si vous avez 1000 classes !!!

Encapsulation de la consultation d'un attribut

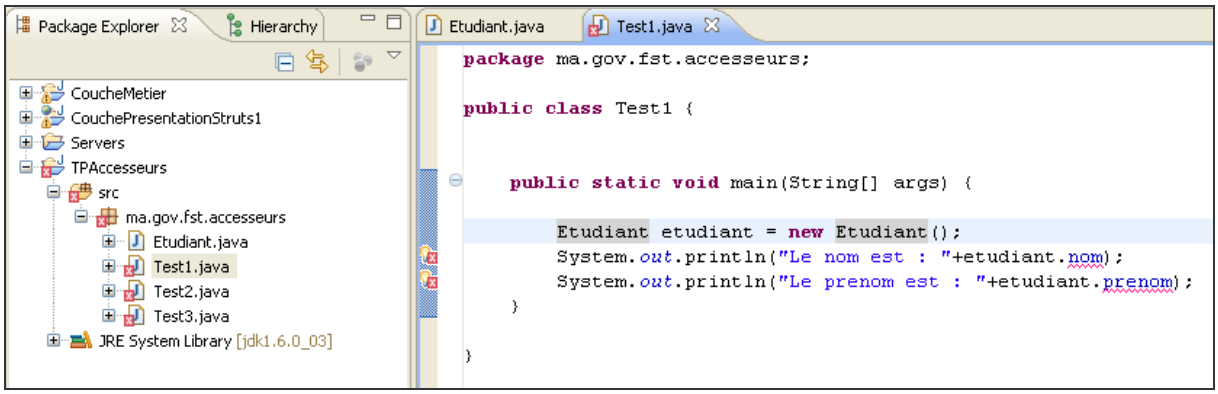
Pour remédier au problème mentionné au niveau de la précédente conclusion, il ne faut pas tolérer un accès direct aux attributs de la classe Etudiant. Autrement dit il faut que les attributs de la classe Etudiant soient déclarés private.

```
public class Etudiant {

      private String nom="Tawssi";
      private String prenom="Said";

}
```

Remarquer que les classes Testx contiennent des erreurs de compilation.



Pour la consultation nous allons créer deux méthodes appelées les accesseurs ou les Getters.

Un Getter est une méthode qui permet d'accéder à la valeur d'un attribut depuis les autres classes. Le nom d'un Getter doit commencer par get suivi par le nom de l'attribut avec la première lettre en majuscule.

Exemple

Nom attribut	Nom Getter
nom	getNom()
prenom	getPrenom()

Vous aurez alors dans la classe Etudiant.

```
package ma.gov.fst.accesseurs;

public class Etudiant {

	private String nom="Tawssi";
	private String prenom="Said";

	public String getNom()
	{
		return this.nom;
	}

	public String getPrenom(){
		return this.prenom;
	}
}
```

Pour accéder aux attributs nom et prénom depuis les classes Testx, il faut passer par les accesseurs (Les méthodes Getters)

La classe Test1 :

```
package ma.gov.fst.accesseurs;

public class Test1 {


	public static void main(String[] args) {

		Etudiant etudiant = new Etudiant();
		System.out.println(etudiant.getNom());
		System.out.println(etudiant.getPrenom());
	}

}
```

La classe Test2 :

```
package ma.gov.fst.accesseurs;

public class Test2 {
	public static void main(String[] args) {
		Etudiant etudiant = new Etudiant();
		System.out.println(etudiant.getNom());
		System.out.println(etudiant.getPrenom());
	}
}
```

La classe Test3 :

```
package ma.gov.fst.accesseurs;

public class Test3 {
	public static void main(String[] args) {
		Etudiant etudiant = new Etudiant();
		System.out.println(etudiant.getNom());
		System.out.println(etudiant.getPrenom());
	}

}
```

Faites l'exécution des classes Testx.

Si maintenant vous voulez changer l'affichage des attributs en les précédant pas les chaines « Le nom est : » et « le prénom est : » vous allez changer uniquement au niveau des Getters de la classe Etudiant.

```
public class Etudiant {

	private String nom="Tawssi";
	private String prenom="Said";

	public String getNom()
	{
		return "Le nom est : " +this.nom;
	}

	public String getPrenom(){
		return "Le prenom est :" +this.prenom;
	}
}
```

Les classes Testx ne sont pas impactées par ce changement.

On dit que la consultation des attributs de la classe Etudiant est encapsulée au niveau des accesseurs. C'est le principe de *l'encapsulation*

Problématique de la modification d'un attribut

On remet les attributs de la classe Etudiant en public.

Maintenant nous souhaitons modifier les attributs nom et prénom de l'objet etudiant au niveau des classes Testx. Mais avec les deux conditions suivantes :

- ✓ *Modifier le nom s'il est différent de la chaine « KADIRI »*
- ✓ *Modifier le prénom s'il est différent de la chaine « Mohammed »*

Vous aurez alors au niveau des classes Testx :

Pour la classe Test1

```
public class Test1 {

	public static void main(String[] args) {
		Etudiant etudiant = new Etudiant();

		if(!"KADIRI".equals(etudiant.nom)){
			etudiant.nom="SALLAMI";
		}

		if(!"Mohammed".equals(etudiant.prenom)){
			etudiant.prenom="SAID";
```

```
            }

            System.out.println("Le nom est : "+etudiant.nom);
            System.out.println("Le prenom est : "+etudiant.prenom);
      }

}
```

Pour la classe Test2

```
public class Test2 {
      public static void main(String[] args) {
            Etudiant etudiant = new Etudiant();
            if(!"KADIRI".equals(etudiant.nom)){
                  etudiant.nom="SALLAMI";
            }

            if(!"Mohammed".equals(etudiant.prenom)){
                  etudiant.prenom="SAID";
            }
            System.out.println("Le nom est : "+etudiant.nom);
            System.out.println("Le prenom est : "+etudiant.prenom);
      }

}
```

Pour la classe Test3

```
public class Test3 {
      public static void main(String[] args) {
            Etudiant etudiant = new Etudiant();
            if(!"KADIRI".equals(etudiant.nom)){
                  etudiant.nom="SALLAMI";
            }

            if(!"Mohammed".equals(etudiant.prenom)){
                  etudiant.prenom="SAID";
            }
            System.out.println("Le nom est : "+etudiant.nom);
            System.out.println("Le prenom est : "+etudiant.prenom);
      }

}
```

Faire l'exécution de vos classes Testx.

Imaginons maintenant que les deux règles de gestion on été changées :

- ✓ *Modifier le nom s'il est différent de la chaine « **FATIMI** »*
- ✓ *Modifier le prénom s'il est différent de la chaine « **SAID** »*

Conclusion

Vous serez obligé de parcourir toutes les classes Testx pour modifier les conditions associées à vos règles de gestion. Imaginer que vous avez 1000 classes !!! La charge de travail sera sans doute énorme.

Encapsulation de la modification d'un attribut

Pour remédier au problème mentionné au niveau de la précédente conclusion, il ne faut pas tolérer un accès direct aux attributs de la classe Etudiant. Autrement dit il faut que les attributs de la classe Etudiant soient déclarés private.

Pour la modification des attributs de la classe Etudiant nous allons créer deux méthodes appelées les Setters.

Un Setter est une méthode qui permet de modifier la valeur d'un attribut depuis les autres classes. Le nom d'un Setter doit commencer par *set* suivi par le nom de l'attribut avec la première lettre en majuscule.

Exemple

Nom attribut	Nom Getter
nom	setNom()
prenom	setPrenom()

Vous aurez alors dans la classe Etudiant.

```
package ma.gov.fst.accesseurs;

public class Etudiant {

	private String nom = "Tawssi";
	private String prenom = "Said";

	public String getNom() {
		return "Le nom est : " + this.nom;
	}

	public String getPrenom() {
		return "Le prenom est :" + this.prenom;
```

```
	}

	public void setNom(String nom) {
		this.nom = nom;
		;
	}

	public void setPrenom(String prenom) {
		this.prenom = prenom;
	}
}
```

Pour modifier les valeurs des attributs nom et prénom depuis les classes Testx, il faut passer par les méthodes Setters.

Pour Test1 :

```
package ma.gov.fst.accesseurs;

public class Test1 {

	public static void main(String[] args) {
		Etudiant etudiant = new Etudiant();
		etudiant.setNom("SALLAMI");
		etudiant.setPrenom("Mohammed");
		System.out.println(etudiant.getNom());
		System.out.println(etudiant.getPrenom());
	}
}
```

Pour Test2 :

```
package ma.gov.fst.accesseurs;

public class Test2 {
	public static void main(String[] args) {
		Etudiant etudiant = new Etudiant();
		etudiant.setNom("SALLAMI");
		etudiant.setPrenom("Mohammed");
		System.out.println(etudiant.getNom());
		System.out.println(etudiant.getPrenom());
	}
}
```

Pour Test3 :

```
package ma.gov.fst.accesseurs;

public class Test3 {
	public static void main(String[] args) {
		Etudiant etudiant = new Etudiant();
		etudiant.setNom("SALLAMI");
		etudiant.setPrenom("Mohammed");

		System.out.println(etudiant.getNom());
```

```
            System.out.println(etudiant.getPrenom());
      }
}
```

Les règles de gestion doivent maintenant être implémentées au niveau des Setters. Si on reprend les deux règles de gestion précédentes :

- ✓ *Modifier le nom s'il est différent de la chaine « **FATIMI** »*
- ✓ *Modifier le prénom s'il est différent de la chaine « **SAID** »*

Il faut apporter les modifications aux Setters de la classe Etudiant comme suit :

```
package ma.gov.fst.accesseurs;

public class Etudiant {

      private String nom = "Tawssi";
      private String prenom = "Said";

      public String getNom() {
            return "Le nom est : " + this.nom;
      }

      public String getPrenom() {
            return "Le prenom est :" + this.prenom;
      }

      public void setNom(String nom) {
            if(!"FATIMI".equals(this.nom)){
                  this.nom = nom;
            }
      }

      public void setPrenom(String prenom) {
            if(!"SAID".equals(this.prenom)){
                  this.prenom = prenom;
            }

      }
}
```

*A noter que les changements des règles de gestion, doivent être implémentés uniquement au niveau des méthodes Setter. On dit que les traitements de modification sont **encapsulés** au niveau des Setter. Encore une autre fois c'est le principe de **l'encapsulation**.*

FIN DE L'ATELIER