L01B\_CLASSES ET OBJETS EN JAVA-1

Table des matières

[1 Création d’un nouveau projet 1](#_Toc1348390037)

[1.1 Création d’une classe 2](#_Toc549382298)

[1.2 Utilisation d’une classe 3](#_Toc52688751)

[1.3 Exécution du code dans IntelliJ IDEA 5](#_Toc206424702)

[2 Ajout d’un attribut et d’une méthode 5](#_Toc1717496454)

[2.1 Modification de la classe Produit 6](#_Toc52548019)

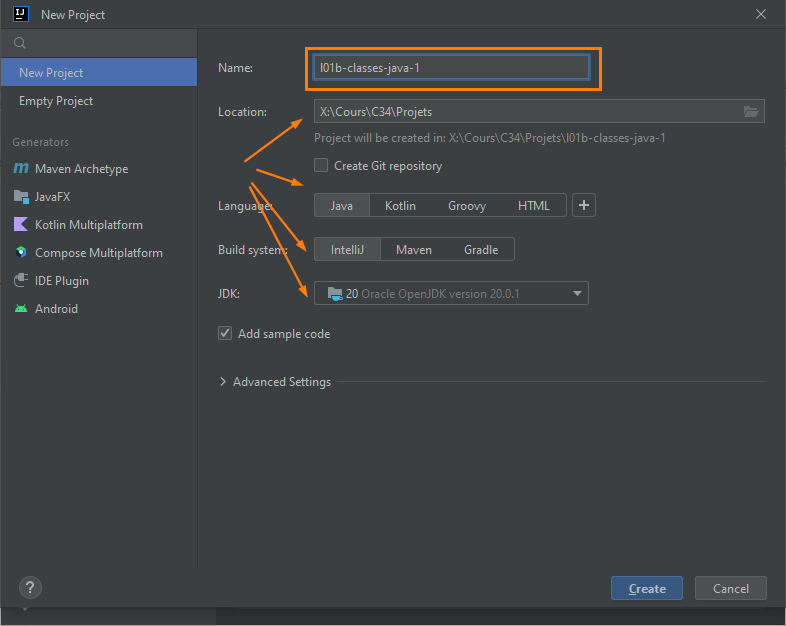
[2.2 Utilisation de la classe Produit modifiée 6](#_Toc943410703)

[3 Structure de fichiers 6](#_Toc380565910)

# Création d’un nouveau projet

L’objectif de ce laboratoire est de voir comment les concepts de POO sont appliqués en Java.

1. Dans IntelliJ IDEA, si vous avez un projet ouvert, fermez-le avec **File** -> **Close Project**, au besoin démarrez IntelliJ IDEA.
2. Créez un nouveau projet nommé l01b-classes-java-1 (les autres paramètres sont les mêmes que dans le premier laboratoire.



Notez qu’on n’a pas créé le répertoire de projet au préalable, le dossier sera créé automatiquement.

## Création d’une classe

1. Faites un clic-droite sur le dossier **src**, puis sélectionnez **New** –> **Java class**.
2. Entrez le nom **Produit** pour la nouvelle classe et cliquez sur **Class**.
3. Un nouveau fichier est créé sous le dossier **src** et l’**éditeur** est ouvert avec un début de code.

NOTE : en Java chaque classe réside dans son propre fichier.

1. Complétez le code dans l’éditeur pour obtenir le résultat suivant :

|  |
| --- |
| public class Produit {  //Declarer que la classe Produit a un attribut nomProduit de type String  public String nomProduit;  //Declarer que la classe Produit a un attribut prix de type double  public double prix;  //Declarer que la classe Produit a une methode nommee AfficherPrix  public void AfficherPrix(){  System.out.println("Le prix de " + this.nomProduit + " est $" + this.prix);  }  } |

Observations et notes additionnelles :

* La déclaration d’une classe requiert le mot clé **class**.
* Les contrôleurs d’accès (public) seront expliqués plus tard.
* Une classe doit toujours utiliser la convention de nom Pascal (Chaque mot du nom commence par majuscule le reste en minuscule). Exemples : Client, Produit, ItemVente, SousAssemblageItem
* Un attribut doit toujours utiliser la convention de nom camelCase (comme Pascal sauf que la première lettre du premier mot est en minuscule). Exemples : nom, prix, coutProduction, descriptionLongueWeb
* Une méthode doit utiliser la convention de nom Pascal.
* Dans une classe la définition d’un attribut est reconnue par le fait que sa syntaxe ressemble à une déclaration de variable.
* Dans une classe, la définition d’une méthode est reconnue par la présence des parenthèses.
* L’expression **this.** Peut être utilisés à l’intérieur d’une méthode pour identifier d’autres éléments du même objet. Elle signifie “cet objet”.

## Utilisation d’une classe

1. Faites un clic-droite sur le dossier **src**, puis sélectionnez **New** –> **Java class**.
2. Entrez le nom **PremiereClasse** pour la nouvelle classe et cliquez sur **Class**.
3. Un nouveau fichier est créé sous le dossier **src** et l’**éditeur** est ouvert dans un deuxième onglet avec un début de code.

Une image contenant texte

Description générée automatiquement

1. Complétez le code dans l’éditeur pour obtenir le résultat suivant :

|  |
| --- |
| public class PremiereClasse {  public static void main(String args[]){  //Declaration d'une variable a de type Produit = creation d'un objet Produit appele a  Produit a = new Produit();  //Assigner des valeurs aux attributs nomProduit et prix de l'objet a  a.nomProduit = "Ordinateur";  a.prix = 1000;  //Utilisation d'un attribut de produit - affichage du pris a l'ecran  System.out.println("Le prix est: $" + a.prix);  //Utilisation d'une methode de produit - affichage d'un message avec le nom et le prix  a.AfficherPrix();  }  } |

Observations et notes additionnelles :

* Une application Java requiert une classe principale (ici appelée PremiereClasse) et une méthode principale appelée obligatoirement main (commence avec une minuscule).
* Pour manipuler un objet on doit utiliser une variable en déclarant sa classe, comme un type. Par la suite, c’est cette variable qui représente l’objet (la variable **a** ici est un objet de la classe produit).
* On accède aux attributs d’un objet avec la notation *objet.attribut* (a.nomProduit, a.prix)
* On accède aux méthodes d’un objet avec la notation *objet.methode()* (a.AfficherPrix())
* On peut assigner une valeur à l’attribut d’un objet avec l’opérateur **=**.
* En POO, une fois les classes définies, leur utilisation rend le code principal très simple. C’est un des avantages de POO.

## Exécution du code dans IntelliJ IDEA

1. Exécutez à partir du fichier PremiereClasse et observez le résultat. Corrigez le code au besoin.

# Ajout d’un attribut et d’une méthode

Complétez les instructions ci-dessous par vous-même.

## Modification de la classe Produit

Modifiez le code existant de la classe **Produit** :

1. Ajoutez un attribut nommé **qteInventaire** de type **int** (commencez la déclaration avec Public…)
2. Ajoutez une méthode nommée **ValeurInventaire** (Commencez la déclaration de la méthode avec Public void…)

La nouvelle méthode doit afficher à la console le message suivant : “La valeur de l’inventaire du produit **X** est $**Y**”

Où **X** est le nom du produit (attribut nomProduit) et **Y** est la multiplication de l’attribut **prix** et de l’attribut **qteInventaire**.

## Utilisation de la classe Produit modifiée

1. Modifiez le code du programme principal (méthode **main** de la classe **PremiereClasse**) :
2. Ajoutez une ligne de code pour assigner la valeur 3 à l’attribut **qteInventaire** de l’objet **a**.
3. Ajoutez une ligne de code pour appeler la méthode **ValeurInvenaire** de l’objet **a**.
4. Exécutez le code, qui devrait afficher le nouveau message avec la quantité d’inventaire en plus des autres messages.
5. Si vous obtenez des erreurs, dépannez et demandez de l’aide au besoin.

# Structure de fichiers

1. Vérifiez votre dossier de projet **l01b-classes-java-1** et observez :

* la présence des 2 fichiers source (.java) dans le dossier **src**
* la présence des 2 fichiers de classe (.class) dans le dossier …\out\production\l01b-classes-java-1\