L01C\_PACKAGES JAVA

Table des matières

[1 Création d’une application avec Packages 1](#_Toc1751387248)

[1.1 Création d’un nouveau projet 2](#_Toc1491789971)

[1.2 Création d’un Package 2](#_Toc635584933)

[1.3 Création des classes à l’intérieur d’un Package 2](#_Toc499819848)

[1.4 Ajout du code du programme principal 4](#_Toc325038657)

[2 Structure de fichiers 6](#_Toc1263699653)

[3 Création et exécution d’un fichier JAR 7](#_Toc38770751)

[3.1 Création du fichier JAR 8](#_Toc1497980130)

[3.2 Exécution du fichier JAR dans Windows 8](#_Toc1108122094)

# Création d’une application avec Packages

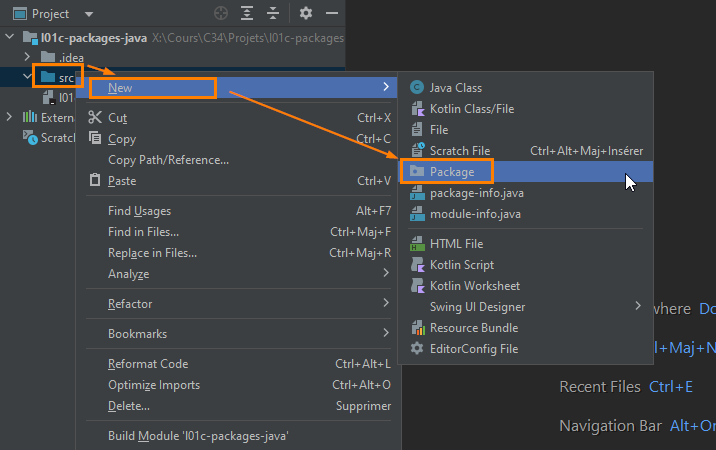
L’objectif de ce laboratoire est de voir comment on créé et on utilise les packages Java, sous IntelliJ IDEA.

## Création d’un nouveau projet

1. Créez un nouveau projet nommé **l01c-packages-java**.

## Création d’un Package

1. Faites un clic-droite sur **src**, sélectionnez **New**, puis **Package**.



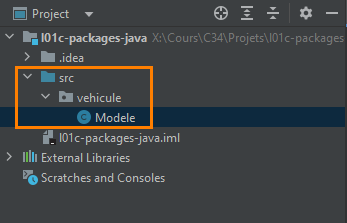
1. Comme nom de package entrez : **vehicule**, puis appuyez sur **Enter**.
2. Observez qu’un nouveau dossier **vehicule** a été créé sous le dossier **src**.

Les conventions de noms pour nommer les packages sont d’utiliser uniquement des minuscules. Exemple : personnes, outilsgraphiques, gestiondesstocks, etc.

## Création des classes à l’intérieur d’un Package

1. Faites un clic-droite sur véhicules et sélectionnez **New** -> **Java class**.
2. Complétez la création de la classe en la nommant **Modele**.
3. Observez que :

* La classe (et son fichier) se trouvent sous le package.
* Le code a automatiquement ajouté une ligne pour indiquer que ce fichier (cette classe) fait partie du package.
* Si on a déjà créé une classe dans son fichier on peut l’ajouter à un package ou la déplacer de package en ajoutant/modifiant cette ligne, qui doit être au début du code.



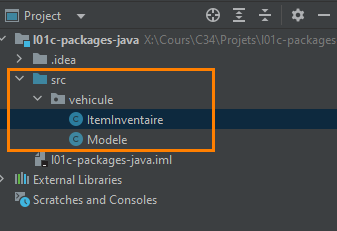
1. Dans la classe Modele, ajoutez :

* Un attribut **idModele** (type String)
* Un attribut **marque** (type String)
* Un attribut **modeleMarque** (type String)
* Une méthode **AfficherInfoModele()** (retourne void) qui affiche à la console une phrase qui inclus les 3 attributs (créez la phrase vous-même).

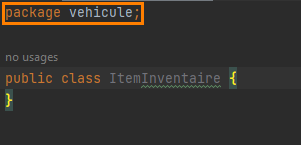
1. Sous le Package **vehicule**, créez une nouvelle classe nommée **ItemInventaire** qui a les caractéristiques suivantes :

* Attribut **idItemInventaire** (type String)
* Attribut **idModele** (type String)
* Attribut **prixSuggere** (type double)
* Méthode **AfficherItemInventaire()** (retourne void) qui affiche à la console une phrase qui inclus les 3 attributs (créez la phrase vous-même).

1. Si vous observez le volet Projet dans IntelliJ IDEA vous devriez voir une structure avec le package vehicule sous le dossier src et 2 fichiers de classes sous **vehicule**.



De plus, chaque classe indique qu’elle est incluse dans le Package **vehicule**, par la première ligne de code.



1. Si vous naviguez dans votre structure de dossiers de Windows, les dossiers doivent refléter la structure affichée dans IntelliJ IDEA, avec le Package **vehicule** comme dossier racine des fichiers de classe, ce qui est différent de la structure sans Packages.

## Ajout du code du programme principal

1. Créez une classe pour le programme principal avec les caractéristiques suivantes :

* Inclus dans le Package vehicule
* Nommée GestionnaireVehicule
* Avec la méthode public static void main()

1. Créez un objet de type **Modele** appelé **m1** et assignez des valeurs aux attributs :

* idModele = **hocrv02**
* marque = **Honda**
* modeleMarque = **CRV-LX**

1. Créez un deuxième objet de type **Modele** appelé **m2** et assignez des valeurs aux attributs :

* idModele = **husan04**
* marque = **Hyundai**
* modeleMarque = **Santa Fe-Urban**

1. Créez un objet de type **ItemInventaire** appelé **iv1** et assignez des valeurs aux attributs :

* idItemInventaire = **aut000011**
* idModele = **hocrv02**
* prixSuggere = **34000**

1. Créez un objet de type **ItemInventaire** appelé **iv2** et assignez des valeurs aux attributs :

* idItemInventaire = **aut320057**
* idModele = **husan04**
* prixSuggere = **38000**

1. Utilisez **println** et les **méthodes** des objets m1, m2 iiv1 et iv2 pour obtenir le résultat ci-dessous :

Une image contenant texte, moniteur, noir

Description générée automatiquement

Le texte dépendra du texte que vous avez créé dans les méthodes de vos classes.

# Structure de fichiers

Lorsqu’on utilise des Packages, la structure de fichiers est adaptée pour tenir compte de ce niveau d’hiérarchie : dans le projet on a des dossiers pour les Packages et les fichiers de classes sont placés à l’intérieur de ces dossiers.

1. Dans la structure de dossiers de Windows, confirmez que les fichiers **.java** du projet ne sont pas directement sous **src**, mais dans un répertoire **vehicule** (dossier du package **vehicule**) qui lui est sous **src**.
2. Vous pouvez vérifier que le même principe s’applique au dossier **out**.

# Création et exécution d’un fichier JAR

Des fichiers **.jar** peuvent être exécutés tout comme des fichiers **.class** par la JVM.

Pour exécuter l’application **GestionnaireVehicule** sur votre station, suivez les instructions ci-dessous :

## Création du fichier JAR

Un Artefact (Artifact) est un assemblage de ressources regroupées pour les tests, le déploiement ou la distribution de l’application.

Un fichier JAR est un artefact.

1. Dans IntelliJ IDEA, avec le projet du l01c ouvert : cliquez sur **File** -> **Project Structure** -> **Artifacts**
2. Cliquez sur **+** -> **JAR** -> **From modules with dependencies**
3. Pour **Main class:**, sélectionnez **GestionnaireVehicule** (la class principale de l’application).
4. Observez que le nom de la classe commence par le nom du package, **vehicule.GestionnaireVehicule**, puis cliquez **OK**
5. Observez la présence du nouvel artefact, in fichier **JAR** dont l’extension est **.jar** et observez l’endroit où sera conservé le fichier : dans le dossier **\out\artifacts\**, puis cliquez sur **OK**
6. La configuration des paramètres du fichier JAR terminée, il faut compiler l’application dans le fichier JAR.
7. À partir du menu : **Build** -> **Build Artifacts…** -> **l01c-packages-java.jar** -> **Build**
8. Observez la présence d’un fichier **MANIFEST.MF** dans un dossier **META-INF** du dossier **src**. Le manifeste contient la liste des fichiers inclus dans l’archive JAR.
9. Observez la présence du fichier **.jar** dans le sous-dossier d’un nouveau dossier **artifacts** sous **out**.

## Exécution du fichier JAR dans Windows

Vous pouvez exécuter un fichier .jar directement dans Windows (dans la mesure où le JRE approprié est installé bien sûr, pour que la JVM soit disponible).

L’application **GestionnaireVehicule** produit un résultat à la console, dans Windows elle doit être exécuté à partir d’une console, donc en invite de commande.

1. Dans Windows, ouvrez une invite de commande entrez les commandes suivantes :

|  |
| --- |
| X:  cd X:\Cours\C34\Projets\l01c-packages-java\out\artifacts\l01c\_packages\_java\_jar\ |

En ajustant le chemin pour refléter votre structure de fichiers.

1. Pour lancer l’application, entrez la commande suivante :

|  |
| --- |
| java -jar l01c-packages-java.jar |

Vous devriez voir l’affichage du message dans la fenêtre d’invite de commande.

Une image contenant texte

Description générée automatiquement