Java
Création d'un exécutable

Compilateur `javac`

- La variable d'environnement CLASSPATH doit contenir les chemins des classes et packages nécessaires à la compilation
- Emploi basique du compilateur `javac`
  - `javac [fichiers_sources]`
- compilation de tous les fichiers sources du répertoire en cours : `javac *.java`
- les fichiers *.class* sont créés, mais ne sont pas déployés dans leur package

Compilateur `javac`

- Pour compiler et déployer
  - `javac -d <repertoire_déploiement> <fichiers_sources>`
  - les répertoires des packages sont alors créés et les fichiers *class* sont déployés dans leur package
  - exemple : `javac -d . *.java`
    - **`-d .`** : déploiement dans le répertoire en cours (.)
- Beaucoup d'autres options sont utilisables sur la commande `javac`
  - reportez-vous à la documentation du JDK

Lancement de l'exécutable

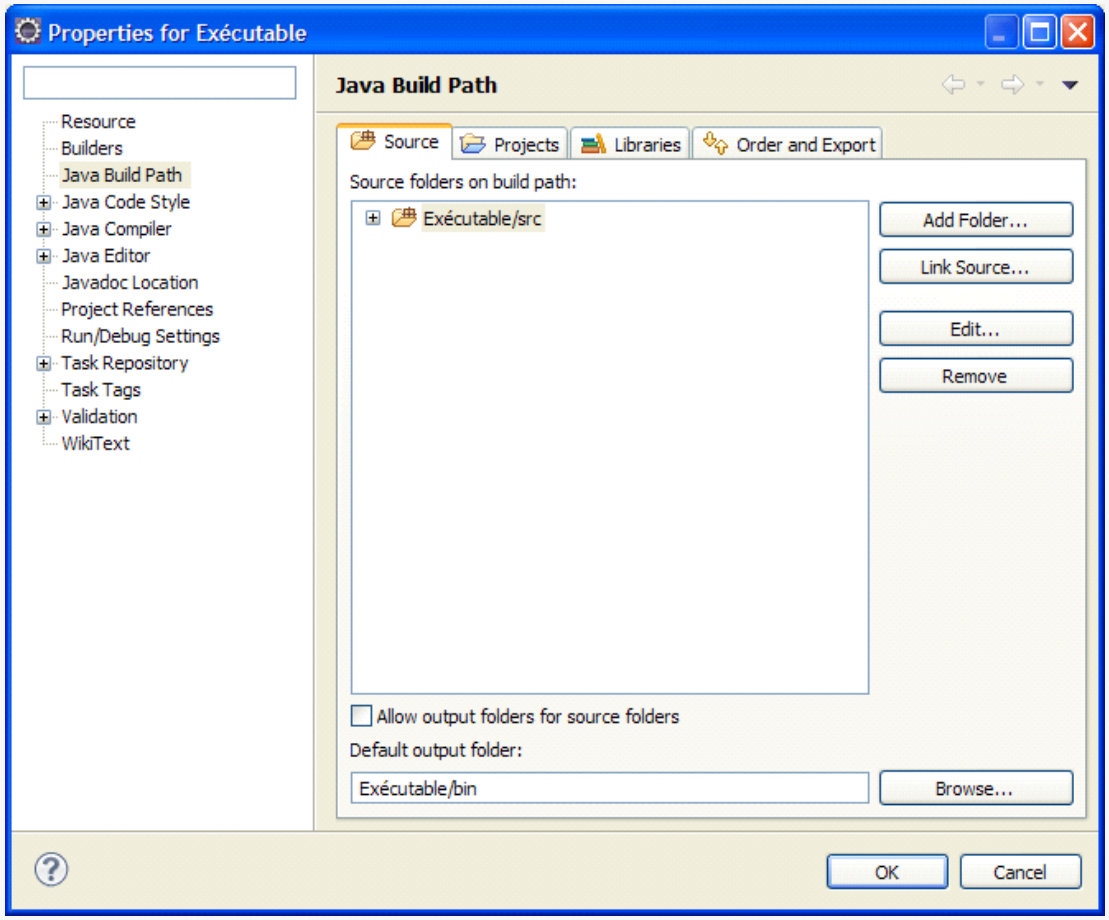
- Le point d'entrée de notre exécutable est la méthode `main` d'une des classes
- Lancement basique de l'exécutable
  - `java package.Classe`
    - ne pas mettre l'extension .class
    - bien préciser le package complet
    - s'assurer que la variable `CLASSPATH` est correcte
- Exemple (`Main` est la classe contenant la méthode `main`) : `java org.antislashn.formation.Main`

Compilation et exécution sous Eclipse

- Eclipse compile à la volée nos classes
  - l'étape de compilation est transparente
    - option du menu **`Project..Build Automatically`** cochée
  - les binaires sont généralement mis dans le répertoire *bin* du projet
    - le répertoire de déploiement n'est pas visible dans la vue *Project*
    - un répertoire différent peut être spécifié dans les propriétés du projet

Compilation et exécution sous Eclipse

- Les propriétés du projet sont accessibles par le menu `Project → Properties`



Compilation et exécution sous Eclipse

- Pour exécuter la classe contenant la méthode main :
  - sélectionnez la classe
  - par un clic droit sur la classe, choisissez **`Run As...Java Application`** dans le menu contextuel
- la console affiche alors le résultat
- une configuration d'exécution est créée

Création d'une archive

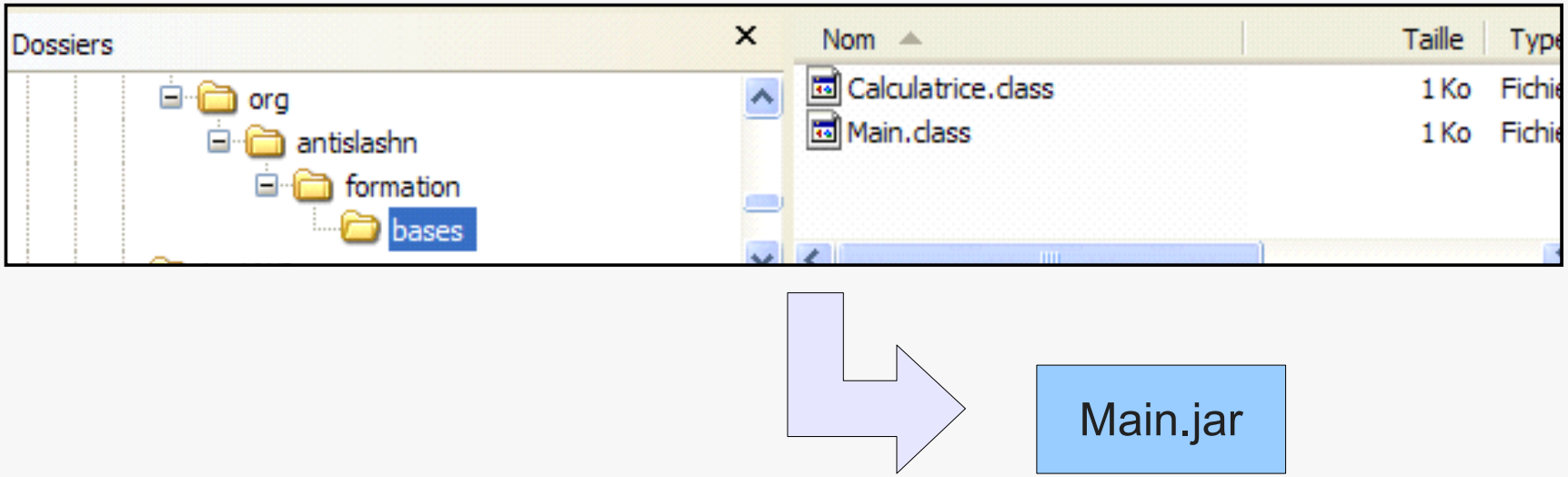
- La distribution d'un exécutable est facilitée par la création d'une archive contenant les classes, les librairies, les ressources de l'application
- L'archive est un fichier compressé au format *zip* dont l'extension est *.jar*
  - peut contenir divers ressources
    - images, fichiers properties, ...
  - contient un fichier META-INF/MANIFEST.MF

Création de l'archive avec `jar`

- L'utilitaire `jar` est fournit avec le JDK
  - `jar cfv <archive> [repertoire/*] [fichiers]`
  - où :
    - c : création de l'archive
    - f : résultat de l'utilitaire jar dirigé vers un fichier
    - v : mode bavard
    - archive : nom du fichier archive
    - répertoire/* : répertoire à inclure dans l'archive, l'inclusion est récursive
    - fichiers : liste des fichiers à inclure, séparés par un espace

Création de l'archive avec `jar`

- Création d'une archive Main.jar
  - `jar cfv Main.jar org\*`
  - l'ensemble des classes et répertoires contenus dans le répertoire *org* sont archivés



Création de l'archive avec `jar`

- L'utilitaire jar à créé un fichier *MANIFEST.MF* qui contient par défaut :

```
Manifest-Version: 1.0
Created-By: 1.6.0_10-beta (Sun Microsystems Inc.)
```

  - ce fichier est dans le répertoire *META-INF* de l'archive
- Le fichier *MANIFEST.MF* doit souvent être modifié, l'utilitaire jar le permet
  - `jar umf ajout-manifest fichier.jar`
  - reportez vous à la documentation du JDK