

Objectifs du laboratoire (2 heures)

1. Le but de ce premier laboratoire est de vous familiariser avec les outils utilisés dans le cours *Informatique 1* et de faire vos premiers pas en programmation *Java* (si ce n'est pas déjà fait !).
2. La durée estimée pour réaliser ce laboratoire est de **deux périodes (1 heure 30 minutes)**.
3. Vous aurez l'occasion durant ce premier laboratoire de créer votre premier projet *Java* et de faire connaissance avec l'environnement de développement *Eclipse*.

Partie 1. Le site web du cours

Le site web du cours sert à assurer l'échange d'informations entre étudiant-e-s et professeur-e-s. Vous y trouverez des informations sur le cours, des anciens examens, les dernières nouvelles... Cette plate-forme est accessible tant à l'intérieur qu'à l'extérieur de la HES-SO Valais. Le site est disponible à l'adresse

<https://infl.begincoding.net>

Tâche 1

Effectuez les tâches suivantes :

1. Essayer de retrouver sur le site le cours du jour ainsi que ce laboratoire.
2. Téléchargez les fichiers `RoomCalc.java` et `Input.java` et retrouvez ces fichiers sur le disque.

Partie 2. Environnement de développement

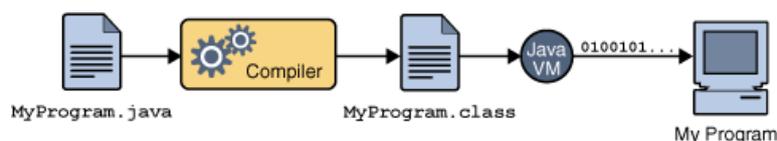
Petite introduction sur le langage Java

Java a été développé par l'entreprise Sun et présenté pour la première fois le 23 mai 1995 comme nouveau langage de programmation orienté objets, simple et indépendant de la plate-forme. Sun (Oracle) possède le droit d'auteur sur le nom Java et garantit que ce nom ne peut être porté que par un langage Java "100% pur". Le langage est toutefois disponible sur tous les systèmes, gratuitement.

Java a pour but de créer un langage de programmation simple et indépendant de la plate-forme, avec lequel on peut programmer non seulement des ordinateurs courants mais aussi des micro-processeurs. Ceux-ci se retrouvent dans des appareils industriels ou ménagers, comme p. ex. les téléphones mobiles, machines à laver, voitures, feux de signalisation, cartes de crédit et systèmes de sécurité, et télévisions dites "intelligentes".

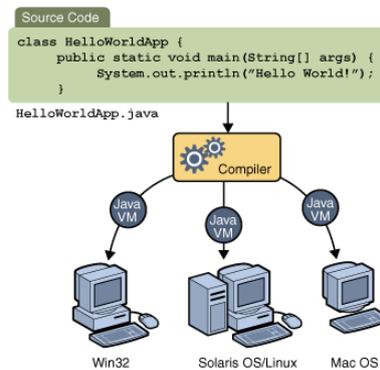
Ce qui a rendu Java populaire est la notion d'applet, petit programme tournant dans un navigateur Web, bien qu'il soit de plus en plus utilisé pour des applications à part entière, y compris des applications de type serveur. Aujourd'hui, Java est surtout utilisé pour réaliser des programmes sur le Web (du côté serveur).

Au niveau de l'exécution du code, les langages de programmation sont soit compilés, soit interprétés. Java combine les deux techniques : le code source est compilé dans un langage intermédiaire, le *bytecode*, qui est ensuite interprété par la machine virtuelle (*Java Virtual Machine, JVM*), comme le montre la figure suivante¹.



C'est ainsi que les programmes Java sont exécutables sur n'importe quelle plate-forme qui implémente la JVM, quel que soit le système qui a produit le *bytecode*.

¹ Les figures sont tirées du tutorial Java d'Oracle.



Les qualités les plus importantes de Java sont :

- Indépendance de la plate-forme
- Orientation objets
- Syntaxe proche du C et du C++
- Vaste bibliothèque de classes

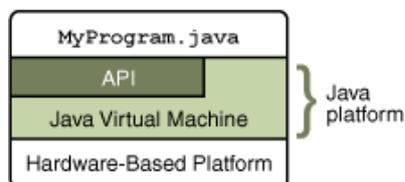
Java VS JavaScript

JavaScript est un langage de script qui peut être intégré à HTML et à la plupart navigateurs Web pour exécuter différentes fonctions et actions. Il ressemble à Java mais est quand même un langage différent. Aujourd'hui, Javascript est très utilisé pour présenter des informations sur le Web (du côté du client) alors que Java est plus utilisé du côté des serveurs.

Le kit de développement Java (Java Standard Development Kit, JDK)

Le kit de développement Java, aussi appelé JDK, comprend deux composants :

- L'environnement d'exécution (*Java Runtime Environment, JRE*), appelé aussi la plate-forme Java, composé de la machine virtuelle et des classes composant la librairie de base (*Application Programming Interface, API*). Si on n'installe sur sa machine que le JRE, on ne peut qu'exécuter des programmes Java, on ne peut pas écrire ses propres programmes.



- Des outils de développement tels que compilateur et débogueur, qui nous permettent d'écrire nos propres programmes.

Java est un langage beaucoup utilisé (même si sa popularité décroît récemment) et qui s'adapte. Il y a ainsi périodiquement de nouvelles versions qui sortent avec des ajouts et des améliorations. La première version du JDK (1) est sortie en 1995 et nous en sommes aujourd'hui à la version 17 (sortie en septembre 2021).

Installation des outils

Pour installer les différents outils, vous retrouverez les instructions sur le site Web du cours ici (<https://inf1.begincoding.net/outils-de-developpement/>). Merci de les suivre scrupuleusement (notamment la version de Java utilisée qui doit être la version 11).

Tâche 2 - Tutoriel Eclipse

Ouvrez la suite de développement **Eclipse**. Ce programme est très utilisé pour écrire des programmes, tant dans le monde académique que dans l'industrie. Il est très puissant mais également très riche (et donc un peu compliqué à prendre en main).

Eclipse utilise la notion de *workspace* (espace de travail) qui représente le travail en cours. Tous les projets que vous ferez se trouvent dans un *workspace*.

- 1) Au démarrage d'Eclipse, choisissez la localisation de votre workspace dans votre répertoire personnel. Par exemple, dans les salles d'informatique sous Linux, si votre acronyme est « pierrean.mudry », mettez comme workspace

```
/home/pierrean.mudry/edu_u/workspace
```

Si vous travaillez sur votre laptop, choisissez un répertoire que vous retrouverez facilement.

- 2) Créez un nouveau projet « **Lab1** » (commande *File* → *New* → *Project*) et créez un projet Java (*Java Project*):
 - a. Introduisez le nom de projet « **Lab1** » dans la zone de texte *Project Name*.
 - b. Vérifier que vous sauvegardez bien votre projet dans le workspace que vous venez de créer et qui se trouve dans votre répertoire personnel.
 - c. Si elle n'est pas sélectionnée, cochez l'option « *Create separate folders for sources and class files* ». Cela aura pour conséquence que les fichiers compilés et les fichiers source se trouveront dans deux chemins différents.
 - d. Appuyez sur *Finish*.
 - e. Selon la version d'Eclipse que vous avez, il se peut que la fenêtre *Create module-info.java* apparaisse. Choisissez l'option *Don't create*. **C'est important !**
- 3) Ajoutez à votre projet une nouvelle classe (commande *File* → *New* → *Class*)
 - a. Introduisez dans la zone de texte « *Name* » le nom de la classe (programme) `MyProgram`. Appuyez sur *Finish*.
- 4) Complétez votre programme pour avoir **exactement** le code suivant :

```
public class MyProgram {
    public static void main(String[] args) {

        // Declares two variables
        int toto = 3;
        int titi = 4;

        // Make the computation
        int theSum = toto + titi;

        // Display the result
        System.out.print("The sum is equal to : ");
        System.out.print(theSum);
    }
}
```

Vous le trouverez aussi dans le fichier `MyProgram.java` sur le site web du cours.

- 5) Par défaut, le code est automatiquement compilé par *Eclipse*. Vous n'avez donc rien de spécial à faire pour voir si votre code comporte des erreurs (qui seront soulignées en rouge).
- 6) Exécutez votre programme
 - a. Cliquez sur la commande *Run* → *Run*.

- b. Vous verrez alors le résultat de votre programme affiché dans la console, à savoir le résultat 7.
 - c. Expérimentez en changeant le programme afin d'avoir une multiplication plutôt qu'une somme.
- 7) Contrôlez les résultats dans la fenêtre de sortie (nommée *Console* au fond de l'écran), et si tout se passe bien, cherchez le fichier `.class` correspondant dans le répertoire de votre projet (les fichiers sources `.java` sont dans le répertoire `src`, tandis que les fichiers compilés `.class` sont dans le répertoire `bin`).

Remarque : vous pouvez compiler et exécuter votre programme en une seule opération à l'aide du bouton **Run**  ou du raccourci clavier `Ctrl+F11`).

Tâche 3 – Ajouter des fichiers

- 1) Copiez les fichiers `RoomCalc.java` et `Input.java` dans le même répertoire que la classe `MyProgram.java`
- 2) Regardez maintenant dans l'explorateur de projet d'Eclipse pour vérifier que `RoomCalc.java` et `Input.java` sont ajoutés dans les fichiers source de votre projet (tapez sur `F5` pour rafraîchir la vue).
- 3) Vous pouvez exécuter le programme `RoomCalc.java` en cliquant avec le bouton droit sur `RoomCalc.java` dans l'explorateur d'Eclipse et en sélectionnant *Run As* → *Java Application*.
- 4) Vous pouvez ensuite cliquer dans la console et entrer les valeurs demandées.
- 5) Essayez de comprendre ce qui se passe dans le fichier `RoomCalc.java`. Bien que nous n'ayons pas vu ces éléments ensemble, vous ne pouvez pas encore comprendre toutes les finesses mais essayez d'imaginer à quoi peuvent servir les différents éléments du programme.
- 6) Modifiez le fichier afin :
- 7) D'afficher le message « `Volume calculator, by John Doe` » (en mettant votre nom à la place de `John Doe`) au début du programme.
- 8) D'afficher le message « `Goodbye and thank you` » à la fin de l'exécution du programme.
- 9) Essayez de calculer et afficher le volume en gallons impériaux (une mesure de volume anglaise). Utilisez pour cela l'information qu'un gallon vaut 0.00454609 m^3 .

Tâche 4 – Votre consommation d'essence

1. Ajoutez une nouvelle classe à votre programme (*File* → *New Class*) que vous nommerez `Fuel`.
2. En vous inspirant de l'exemple ci-dessus, faites un programme permettant de calculer votre consommation d'essence sur une certaine distance. Un exemple possible de ce qui est attendu pourrait-être :

```
Liters / 100 km : 12
Distance driven : 10
You used : 1.2 liters of fuel.
```

Tâche 5 – Flottaison d'une sphère

Nous voudrions écrire un programme qui permet de déterminer si une sphère en métal creuse flotte si on la plonge dans l'eau. Il doit être possible de choisir la matière qui compose la sphère, l'utilisateur devra donc saisir la masse volumique de cette matière (à titre d'exemple, la masse volumique de l'aluminium est de 2.6 g/cm^3).

L'utilisateur doit donc saisir le rayon de la sphère (ici le rayon extérieur), l'épaisseur de la surface et la masse volumique. À partir de ces données, calculez le volume intérieur de la sphère ainsi que la masse de la sphère creuse. Si le rapport entre la masse et le volume est inférieur à 1, on peut considérer que l'objet flotte. Un exemple d'exécution nous donnerait :

```
Please enter outer sphere radius (in cm): 15
Please enter surface thickness (in cm): 1
Total outer sphere volume: 14137.1669411540 cm3
Enter material density (in g/cm3): 2
```

```
Total object density: 0.37392592592592594 => The object is floating
```