Universidade de Aveiro, DETI

Programação Orientada por Objetos

Guião das aulas práticas

Ano: 2020/2021

Prática 1 Aula introdutória

Objetivos

Os objetivos deste trabalho são:

1. Instalar as ferramentas necessárias de desenvolvimento em Java
2. Comparar a estrutura de um programa em Java e outro em Python
3. Perceber a organização de projetos e programas Java
4. Editar, compilar e executar programas em Java (com base nas ferramentas instaladas no ponto 1.)
5. Compilação e execução em linha de comandos (*opcional*)
6. Preparar o projeto Java para a próxima aula

Exercício 1.1 Instalação das ferramentas de desenvolvimento em Java

Existe uma grande probabilidade de já ter o Java instalado no seu computador pessoal. No entanto, trata-se de uma versão composta por um conjunto de componentes (e.g., bibliotecas) que são necessários para executar programas Java no seu computador (i.e., o *Java Runtime Environment* ou JRE). Para construir programas, torna-se necessário instalar uma outra versão, designada por *Java Development Kit* (JDK), que é o sistema de desenvolvimento para Java (contém bibliotecas, compilador, interpretador, etc.). Com o compilador e um editor de texto comum (i.e., Notepad) é já possível escrever e compilar programas em Java, usando a linha de comandos. No entanto, o Java é amplamente utilizado em projetos de alguma dimensão, tornando-se apelativo o uso de ferramentas integradas de desenvolvimento de projetos de software, também conhecidas como *Integrated Development Environment,* ou IDE. Existem múltiplos IDE, tendo como exemplos o Eclipse, Intellij, Netbeans, CLion, Geany, entre outros.

No âmbito desta disciplina, vamos privilegiar o uso do **Visual Studio Code** (VS Code).

Como os IDE não pretendem endereçar nenhuma linguagem de programação em particular, não costumam vir com compiladores pré-instalados. Assim, também é muito comum que os IDE permitam instalar extensões, que os capacitam de interpretar e compilar linguagens selecionadas pelos utilizadores.

No caso do VS Code, existe um pacote pré-concebido, que já engloba o IDE, o JDK e uma extensão ("Coding Pack for Java") para o desenvolvimento em Java. Esse pacote pode ser obtido a partir de:

Windows - https://aka.ms/vscode-java-installer-win
macOS - https://aka.ms/vscode-java-installer-mac
Linux: *Different alternatives (Note that you must install the JDK by yourself according to your distribution and processor architecture)*
- VSCode (only) *deb or .rpm packages*:
  https://code.visualstudio.com/download
- VSCode (only) *distribution-based packages*:
  https://code.visualstudio.com/docs/setup/linux

- "Java Extension Pack"*you can download it from the VS Code in-app marketplace or via* https://marketplace.visualstudio.com/items?itemName=vscjava.vscode-java-pack

Uma vez instalado o programa, podemos ir ao menu **View → Explorer**, e escolher no separador que surge, a opção **Create Java Project**. Na lista que surge, selecionar a opção **No build tools**. Em seguida, surge uma janela do explorador de ficheiros, para podermos indicar onde queremos que o nosso projeto fique alojado. Por fim, surge mais uma caixa de texto para podermos identificar o nome do nosso projeto (e.g., POO).
Do lado esquerdo, irá aparecer uma estrutura de ficheiros com as pastas "lib" e "src". A pasta "lib" irá conter as bibliotecas necessárias ao nosso projeto e a pasta "src" irá conter o código fonte (i.e., ficheiros .java) que terão o nosso código. Poderemos verificar que dentro da pasta "src" existe já um ficheiro "App.java" que tem um exemplo de um programa em Java.

Para compilar e executar o programa podemos ir ao menu **Run → Run Without Debugging**, ou premir simultaneamente as teclas **Ctrl** e **F5** no teclado, ou simplesmente clicar com o botão direito em cima do ficheiro "App.java" e selecionar a opção **Run**. Após essa autorização ser concedida, o programa é compilado e o resultado aparece numa janela de terminal embutida no VS Code, surgindo a frase "Hello, World!"

Para se familiarizar com o VS Code recomenda-se a leitura de algumas fontes. Por exemplo:
https://code.visualstudio.com/docs/introvideos/basics
https://code.visualstudio.com/docs/java/java-tutorial

Exercício 1.2 Comparando Código em Java e Python

Vejamos o código do ficheiro "App.java"

```java
public class App {
   public static void main(String[] args) throws Exception {
      System.out.println("Hello, World!");
   }
}
```

Vejamos agora o equivalente em Python:

```python
def main():
   print("Hello, World!");
```

Aqui pretende-se verificar as diferenças nas sintaxes de Python e Java. Existem muitas outras diferenças, nomeadamente o Python é uma linguagem interpretada, ao passo que o Java precisa de compilar o código fonte e depois executá-lo. Ambos têm diferenças de desempenho também.

Do ponto de vista de sintaxe, tente verificar no código as seguintes diferenças:

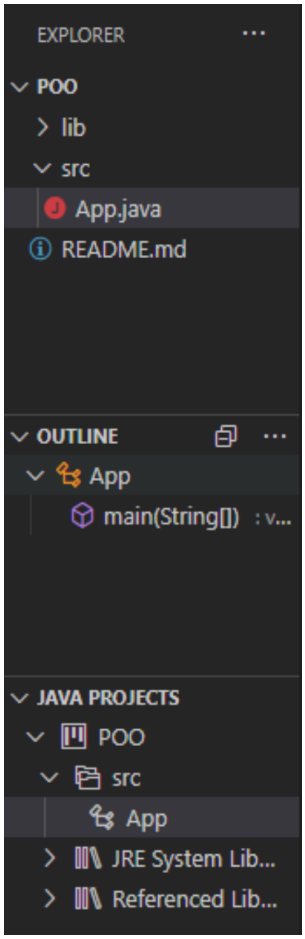
- Um programa em Java precisa de definir uma classe e todo o código está dentro dessa classe. Ou seja, em Java, tudo é um objeto.
- Tudo em Java (funções, variáveis, etc.) pertence a um tipo de dados (e.g., void, int, etc.). No Python, o tipo de dados é dinâmico. Portanto diz-se que o Java é *Strongly Typed* e o Python é *Weakly Typed*. Isto fica mais claro neste exemplo de Java:

```java
public class App {
 public static void main(String[] args) throws Exception {
    String frase = "Hello, World!";
    System.out.println(frase);
 }
}
```

- Um programa em Java precisa de ter uma função chamada `public static void main(String[] args);`
- A sintaxe do Python assenta em indentação. No Java, a indentação não tem valor: o âmbito de cada instrução (e.g., função, ciclo, classe) é definido dentro de um conjunto de " { " e "}". Diz-se que o que estiver compreendido entre os "{….}" está dentro de um *bloco*. Adicionalmente, cada expressão em Java é finalizada por um ";". Se estes elementos falharem, há um erro de sintaxe sinalizado pelo compilador.

Mais detalhes sobre as diferenças de sintaxe entre ambas as linguagens podem ser verificados no ficheiro `Python-Java-Comparison.pdf` disponível no elearning.

Exercício 1.3 Perceber a organização de projetos e programas Java



No VS Code (com as extensões provenientes do pacote instalado anteriormente), tem três áreas de interação com o projeto, apresentadas na figura à esquerda. A primeira, *EXPLORER*, mostra-lhe a estrutura de ficheiros na pasta (i.e., *Workspace*) onde tem o código do seu projeto. A segunda, *OUTLINE*, apresenta de forma contextual os métodos e elementos da classe apresentada no editor. A terceira, *JAVA PROJECTS*, permite-lhe gerir o projeto Java.

Vamos criar uma nova classe no projeto existente "POO". Para isso, na componente "JAVA PROJECTS", passe com o ponteiro do rato a pasta "src", onde irá depois aparecer o símbolo de adicionar "+". Clique nesse símbolo e selecione a opção "Create New Class". Crie uma classe chamada "MyFirstClass" (sem as aspas).

a) Escreva no editor o seguinte código e execute o programa.

```java
public class MyFirstClass {

   public static void main(String[] args) {
          System.out.println("Hello VS Code!");
   }
}
```

b) Modifique o código de acordo com o exemplo seguinte e execute. Analise o seu funcionamento. Faça outras alterações ao programa (valores, operações, ...) e verifique erros/resultados.

```java
public class MyFirstClass {

   public static void main(String[] args) {
      System.out.println("Hello VS Code!");
      int sum = 0;
      for (int i = 1; i <= 100; i++) {
         sum += i;
```

```
        }
        System.out.println(sum);
    }
}
```

c) Fora do VS Code, abra o programa de gestão de ficheiros (*Explorer*, *File manager*, *Finder*, ...) e verifique a estrutura de pastas e ficheiros que criou até agora. Que pastas existem? Para que são usadas? Compare com a estrutura do "*EXPLORER*" no VS Code.

Exercício 1.4 Compilação e execução em linha de comandos

Nota: Este exercício é opcional.
O pacote instalado no ponto 1, além de instalar o VS Code e as extensões Java, também instala o JDK (neste caso, a versão 11). Isto significa que temos acesso ao compilador através da linha de comandos (e não só através do VS Code).

Após a instalação do JDK pode começar a escrever e executar programas, com o auxílio de um editor de texto simples, o compilador (*javac*) e o interpretador (*java*). Estes três programas podem ser usados através da linha de comandos.

Utilizando um editor de texto qualquer (*vim*, *notepad*, *etc.*), crie o ficheiro `Hello.java` com o conteúdo seguinte:

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("O nosso primeiro programa!");
    }
}
```

a) Usando a linha de comando (*Terminal* em Linux, *COM* em Windows, etc.), compile este ficheiro utilizando o comando: `javac Hello.java`

   Nota: se tiver problemas em executar os programas javac e java, configure a variável de ambiente PATH para que indique ao sistema operativo a localização do compilador de Java (procure soluções online dependendo do sistema operativo que estiver a usar).

   Depois de compilar o código, certifique-se de que foi criado o ficheiro `Hello.class` na mesma pasta.

b) Utilize o comando java para executar o programa criado: `java Hello`
   Certifique-se de que o programa faz o pretendido.

Exercício 1.5 Outros exemplos de código

Na pasta aula01 do *elearning* estão vários ficheiros java. Analise sumariamente cada programa, execute e verifique o seu resultado.

Exercício 1.6 Preparar o projeto Java para a próxima aula (Aula2)

Tendo em conta o procedimento realizado no Exercício 1.3, na componente "EXPLORER", passe o ponteiro do rato sobre o nome pasta "src" e das 4 opções possíveis que surgem, escolher a de adicionar uma nova pasta , dando-lhe o nome de "Aula2".

Passe o ponteiro do rato sobre a pasta "Aula2", selecione agora a opção de adicionar um novo ficheiro e dê-lhe o nome de "Ex1.java" (i.e., ficheiro de código Java do primeiro exercício do Guião 2).
Um modelo básico de código é automaticamente produzido pelo VS Code no editor de texto. Repare que agora a primeira linha desse programa indica "package Aula2;". Isto significa que todas as classes que produzir nesta "pasta" pertencem a um "pacote" (i.e., conjunto) de classes Java que, neste caso, estão associados à "Aula2". Este mecanismo permite assim aceder a classes de umas aulas para as outras.
Na próxima aula prática, poderá continuar neste ficheiro, criando novas classes para cada um dos exercícios desse guião, e assim sucessivamente.