Programação Orientada a Objetos
SANTOS, Rafael

Capítulo 3 – Aplicações em Java

Método main

- Nome **main**
- Método **de classe**: **static**
- Parâmetro: **String**[]
- Sem retorno: **void**
- Modificador de acesso: **public**
- Ver classe MaisDemoData (cap. 3)

```
public static void main(String[] args) {
    ...
}
```

Palavra-chave new

- **Cria** uma instância (objeto) de uma classe
- Aciona o **construtor**
- Ver classes:
  - Ponto2D (Cap. 3)
  - DemoPonto2D (Cap. 3)

Método toString()

- Trata-se de um método herdado da classe Object (será estudada no capítulo 9)
- Retorna uma cadeia de caracteres contendo uma representação do estado de um objeto

```
public String toString() {
    return ...;
}
```

Palavra-chave null

- Valor especial que representa um **nulo**
- Serve para **inicializar** uma referência sem que ela aponte para qualquer objeto
- Permite a **compilação**, mas tem o potencial de gerar uma **NullPointerException**.
- Ver classe:
  - DemoReferencias (Cap. 3)

Entrada de dados com Scanner

- Para usá-la, devemos importar:
  - import java.util.Scanner;
- Criação do objeto da classe Scanner
  - Scanner <nome_ref> = new Scanner(System.in) ;
- Possui métodos para receber alguns os tipos primitivos: next<tipo>()
- O método nextLine() recebe uma linha completa até CR (para usá-lo após next<tipo>(), deve-se limpar o buffer do teclado com outro nextLine())
- Ver classe: DemoPonto2DScanner (Cap. 3 JE)

Exercício de fixação

- Até agora, foram criadas classes de teste (com o método main) onde objetos eram instanciados, suas variáveis de instância eram inicializadas com valores literais e depois exibidas. Crie situações semelhantes, substituindo os valores literais por entradas de dados, usando a classe Scanner. Lembre-se de elaborar interfaces amigáveis com o usuário!