Programación 2

Elementos Adicionales de Programación Básica en Java

Profesores:
Ismael Figueroa - ifigueroap@gmail.com
Eduardo Godoy - eduardo.gl@gmail.com

¿Qué es un método estático?

En Java los atributos y los métodos están asociados por defecto con *instancias de una clase*. Esto significa que deben ser invocados desde una instancia, por ejemplo `auto.acelerar(10)` ...

Pero a veces es deseable y/o necesario asociar métodos *directamente a una clase*:

- Métodos utilitarios: por ejemplo un método `millasAKilometros`, no tiene por qué estar asociado a una instancia, es algo general en el ámbito de los autos.
- Si el método en realidad no ocupa ninguna variable de instancia...

Métodos Estáticos vs No-Estáticos

```java
public class AutoEst {
  String marca;
  int agno;

  public static double
  millasAKm(double m) {
    return m * 1.60934;
  }
}

public class Auto {
  String marca;
  int agno;

  public double
  millasAKm(double m) {
    return m * 1.60934;
  }
}
```

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

    // Quiero calcular 102
    // 102 millas a KM

    // Con método no estático:
    System.out.println(
        new Auto().millasAKm(102));

    // Con método estático:
    System.out.println(
        AutoEst.millasAKm(102));
  }
}
```

¿Qué es un arreglo?

Un arreglo es una colección secuencial de elementos. En Java los arreglos pueden ser estáticos o dinámicos, y todos los elementos deben tener el mismo tipo de dato.

En general los pasos para trabajar con arreglos son:

- Declarar una variable para hacer referencia al arreglo
- Crear o instanciar un nuevo arreglo y asignarlo a la variable recién declarada
- Almacenar, actualizar, manipular los valores en el arreglo

Arreglos Estáticos

- Tienen un largo fijo determinado en su creación.
- Los arreglos están indexados por número, desde el número 0 en adelante.
- Un arreglo de elementos de tipo `T` se escribe `T[]`, y se pueden inicializar directamente, o construir usando `new`.

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

    String[] nombres = { "P", "J", "D" };
    double[] notas   = new double[3];

    notas[0] = 4.2;
    notas[1] = 7.0;
    notas[2] = 6.6;

    System.out.println(nombres.length);
    System.out.println(nombres);
    System.out.println(notas);
  }
}
```

Propiedades Clave

- El acceso a la posición `i` del arreglo a se escribe `a[i]`.
- Los arreglos son un objeto bastante simple, que tiene como principal propiedad su largo.
- El largo está almacenado en la propiedad `length`, la cual se define en la construcción del arreglo.

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

    String[] nombres = { "P", "J", "D" };
    double[] notas   = new double[3];

    notas[0] = 4.2;
    notas[1] = 7.0;
    notas[2] = 6.6;

    for(int i=0; i < notas.length; i++) {
      System.out.println(
      "N[" + i + "]: " + notas[i]);
    }
  }
}
```

La clase utilitaria java.util.Arrays

- La clase `java.util.Arrays` contiene muchos métodos utilitarios para la manipulación de arreglos estáticos.
- La clase tiene solamente *métodos estáticos* por lo que no es necesario tener una instancia—sólo se llaman los métodos.
- Como ejemplo, veamos los métodos `fill`, `sort`, y `toString`.

```
import java.util.Arrays;

public class Main {
  public static void main(String[] args) {
    double[] notas   = new double[10];
    // Rellenamos con valor 3.3
    Arrays.fill(notas, 3.3);

    // Imprimimos representacion en String
    System.out.println(
        Arrays.toString(notas));

    // Modificamos un valor y ordenamos
    notas[9] = 0; Arrays.sort(notas);

    System.out.println(
        Arrays.toString(notas));
  }
}
```

Arreglos Dinámicos

- Tienen un largo variable, que se ajusta automáticamente a medida que se agregan o eliminan elementos.
- Los elementos siguen estando indexados por posición desde el número 0 en adelante.
- En Java corresponden a la implementación de la interface `List`, que representa una colección ordenada de elementos.
- La implementación más usada es la clase `ArrayList`

```
import java.util.List;
import java.util.ArrayList;

public class Main {
  public static void main(String[] args) {
    List<String> nombres =
        new ArrayList<String>();

    nombres.add("P");
    nombres.add("J");
    nombres.add("D");

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

Métodos Clave para Listas[1]

- `add`: agrega un elemento al final de la lista, o en una posición específica dada
- `contains`: retorna un booleano indicando si un elemento pertenece o no a la lista
- `get`: retorna el elemento que está en una posición dada de la lista
- `isEmpty`: retorna verdadero si la lista no tiene elementos
- `size`: retorna el tamaño de la lista
- `indexOf`: retorna el índice del primer elemento que es igual al dado como parámetro, o retorna `-1` si no hay
- `remove`: remueve el elemento en la posición dada de la lista
- `subList`: retorna una nueva lista que corresponde al segmento dado como parámetro
- `toArray`: retorna un arreglo estático con los elementos de la lista

[1] `https://docs.oracle.com/javase/8/docs/api/java/util/List.html`

Iteración sobre una Lista

- La interface `List` es una colección *iterable*...
- Por lo tanto se puede usar el constructo `for` especial para recorrer los elementos de la lista, sin preocuparnos por el largo

```
import java.util.List;
import java.util.ArrayList;

public class Main {
  public static void main(String[] args) {
    List<String> nombres =
        new ArrayList<String>();

    nombres.add("P");
    nombres.add("J");
    nombres.add("D");

    for(String n : nombres) {
      System.out.println(n);
    }
  }
}
```

Parámetros desde Línea de Comandos

- El método `main` toma obligatoriamente como argumento un valor `String[] args`
- Ahora sabemos que eso es un arreglo estático. Haga un programa que imprima sus valores...
- El arreglo toma sus valores de los parámetros usados en la invocación del programa desde la línea de comandos. Por ejemplo:

```
> java Main hola 123
```

La Clase Object[2]

La clase Object está en la raíz de la jerarquía de herencia de Java. Todas las clases heredan, directa o indirectamente, desde Object. Esta clase representa el tipo de dato más general posible.

[2] https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html

El método toString

- La clase Object define el métod toString, el cual puede ser sobre-escrito por las clases descendientes...
- Esto es útil para imprimir información relevante sobre nuestras propias clases, ya que hay muchos lugares donde se invoca implícitamente el método toString

```
public class Auto {
  String marca;
  int agno;

  public Auto(String m, int a) {
    marca = m;
    agno = a,
  }

  public String toString() {
    return "Auto m:" + marca + " a:" + agno;
  }
}
```

Preguntas

Preguntas?